

网管实战宝典

专业网管笔记　成就资深网管

Linux服务器架设与管理

伍之昂　汤　楠　庄　毅◎主编

PROFESSIONAL NETWORK MANAGEMENT
SENIOR NOTES ACHIEVEMENTS

清华大学出版社

网管实战宝典系列丛书

Linux 服务器架设与管理

伍之昂　汤楠　庄毅　主　编

清华大学出版社

北　京

内 容 简 介

本书系统、全面地介绍了 Linux 系统中服务器的架设和配置方法，全书共分 16 章，内容包括 Linux 简介和安装、Linux 基本网络配置、Linux 防火墙、远程控制、NFS 和 NIS、DHCP 服务器、Web 服务器、FTP 服务器、电子邮件服务器、MySQL 数据库、LDAP 目录服务、Samba 服务器、网络时间服务器、网络服务器监控等。

本书在介绍每种 Linux 服务器的架设之前，首先介绍相关的基础理论，然后在实际的场景中介绍 Linux 服务器的架设和配置，以具体问题的求解为导向，以便于读者掌握具体章节的重点及提高实际操作能力。本书结构清晰、易教易学、实例丰富、可操作性强、学以致用，对易混淆和实用性强的内容进行了重点提示和讲解，可作为大中专院校的教材和各类培训班的教材，也适合网络管理员及使用 Linux 的科技人员参考阅读。

本书配有光盘，光盘中附有视频教学软件，并提供架设 Linux 服务器的常用软件，以及一些使用 shell 脚本和程序源代码。

图书在版编目(CIP)数据

Linux 服务器架设与管理/伍之昂，汤楠，庄毅主编. —北京：清华大学出版社，2008.8
(网管实战宝典)
ISBN 978-7-302-18066-1

Ⅰ.L…　Ⅱ. ①伍…　②汤…　③庄…　Ⅲ. Linux 操作系统　Ⅳ. TP316.89

中国版本图书馆 CIP 数据核字(2008)第 098280 号

责任编辑：章忆文　闫光龙
封面设计：柏拉图+创意机构
版式设计：北京东方人华科技有限公司
责任校对：李玉萍
责任印制：

出版发行：清华大学出版社　　**地　址：**北京清华大学学研大厦 A 座
http://www.tup.com.cn　　**邮　编：**100084
社 总 机：010-62770175　　**邮　购：**010-62786544
投稿与读者服务：010-62776969, c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015, zhiliang@tup.tsinghua.edu.cn
印 刷 者：
装 订 者：
经　销：全国新华书店
开　本：185×260　**印　张：**24.25　**字　数：**583 千字
版　次：2008 年 8 月第 1 版　**印　次：**2008 年 8 月第 1 次印刷
印　数：1~4000
定　价：38.00 元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题，请与清华大学出版社出版部联系调换。联系电话：010-62770177 转 3103　　产品编号：

丛 书 序

他山之石，可以攻玉。——诗经

你应该了解真相，真相会使你自由。——圣经

策划初衷

网管员(Network Administrator)是国家劳动和社会保障部近年颁布的第四批国家职业标准中明确规定的一个新兴职业。网管员职业要求从业者具备一系列专业、高端的计算机及网络操作技能。

为了给广大网管员提供一套标准实用的高效实战教材，清华大学出版社在广泛调研与充分论证的基础上，聘请了国内著名院校资深学者和实战经验丰富的网管专家，历时 18 个月精心打造了这套《网管实战宝典》系列丛书。本丛书由网管员的职业应用切入，根据网管员的行业内容细划科目，以实际工作的项目案例为主线，解决实际应用中可能出现的问题，是目前市面上惟一从“网管员职业应用案例实战”角度切入的精品丛书。本套丛书全面介绍网络管理、设计与维护的热点应用案例，剖析透彻，确保技术的先进性、实用性和深入性，是网络管理员必备的实践读物。

首推书目

本套丛书首批推出 10 本。

(1) 家庭、宿舍、小型企业、商务、网吧局域网一本通
(2) 中型企业局域网一本通
(3) 网络规划、设计与配置
(4) Windows Server 2003 配置与管理
(5) Windows Server 2003 服务器架设与管理
(6) 网络管理工具使用大全
(7) 网络安全大全
(8) Linux 服务器架设与管理
(9) 网络故障排除与维护技巧(Windows 版)
(10) 网络故障排除与维护技巧(Linux 版)

丛书特色

本丛书具有以下主要特色。

1．针对性

从网管员职业应用切入，以网管员的行业内容细划科目，所介绍的内容紧紧围绕网管员必备的知识与技能展开，从而突出针对性。

2．实用性

以实际的项目案例为主线，解决实际应用中可能出现的问题，不仅仅是理论上的介绍。这些应用案例是专业人士多年的网管实战经验总结，对读者有最直接、最宝贵的指导意义。

3．可操作性

本丛书在介绍各种实际应用配置方案时，都以图解、截屏等方式与清晰的步骤相结合，并着重强调了各步的配置细节，方便读者按步骤操作，以便快速掌握案例操作过程。

4．先进性

本丛书所介绍的各种网络技术和方案均是当前最主流或最新的，读者通过阅读本丛书即可了解符合实际的网络技术与应用方案。

5．深入性

从书中的应用案例讲解细致入微，分析透彻，过程完整，从而能够确保读者完全理解与掌握，以便在实际工作中应用和借鉴。

6．精彩点拨

从书以大量点评与拓展、注意、提示等特色段落为辅助，帮助读者加深理解关键技术，使读者学得轻松、记得深刻、用得灵活。

7．配视频光盘

每本图书都配有多媒体光盘，光盘将案例的操作步骤录制成视频教学软件，读者只需根据视频教学软件即可自己动手实践操作，迅速提高动手能力和技术水平。同时，丛书将图书中相关的配置文件一并附在光盘之中。

读者对象

这套《网管员实战宝典》可作为有志于从事网络管理职业的读者自学实践教材，也可作为大专院校计算机相关专业的教材或相关师生的参考书以及网络培训教材。



创作团队

我们一直深信：一流的团队可以奉献一流的作品，成就一流的读者。本丛书创作团队来源于著名院校资深学者、实战经验丰富的网管专家，他们长期工作于网管一线，有多年的网络管理与设计经历，经验丰富，实力雄厚。

演示光盘

丛书中应用案例的操作过程以视频的方式实录在光盘中，如同专家亲临现场，手把手教会读者网络服务的搭建和网络设备的配置等操作。另外，光盘中还含有相关的实用资源包。

互动交流

读者的进步就是我们的心愿。本丛书愿为读者提供全面的技术支持，服务方式包括以下方面。

- 技术讲座。将在适当的时间组织专家进行技术巡讲，介绍最新的技术并当面解答读者的疑问。
- 版本升级。本丛书将跟踪最新网络技术发展动态，及时更新版本，为读者提供最新的网络技术。
- 问题解答。如果您在阅读本丛书过程中发现任何问题或疑问，或有意见和建议，请发邮件至 Book21Press@126.com，我们将及时地为您提供解决方案。

特别致谢

在此，我们对丛书所选用的参考文献的著作者，及丛书所引用网站资源及其他相关资源的著作者表示真诚的感谢，并感谢为本丛书出版提供帮助的各界人士。

知识是一座宝库，而实践是打开宝库的钥匙。
从别人成功的经验中学习，是获取知识的捷径。
我们乐意与您一同分享成功的网管实践经验。

——丛书编委会

前　言

当今时代是网络的时代，以 Internet 技术为基础的计算机网络已成为人们生活中最重要的基础设施之一，因此如何在互联网上架设各种功能的服务器成为一项实用而必需的技术。Linux 作为一种免费的操作系统，以其稳定的性能、开源的理念逐步成为服务器操作系统的主流。Fedora 作为继 RedHat 9 之后的 Linux 版本，以其更好的开放性、创新性和前瞻性成为服务器操作系统的首选版本，本书旨在系统地介绍 Fedora Core 6 平台下各种网络服务器的架设和配置。

1．本书阅读指南

第 1 章介绍 Linux 的起源、Linux 的优点、Fedora 与 Linux 的关系等基本知识，并重点介绍 Fedora 6 的安装过程，最后介绍后续章节需要用到的 Linux 系统的一些基本知识。

第 2 章以 TCP/IP 协议族为主线回顾了网络基础知识，然后介绍 Linux 连接 Internet 的基本网络配置，以及如何在单网卡的服务器上架设路由。

第 3 章介绍防火墙的基本知识，重点介绍如何使用 iptables 防火墙软件来架设 Linux 包过滤防火墙，以及如何配置 NAT 服务以实现共享上网。

第 4 章主要介绍服务器远程控制问题，并详细地介绍目前三种远程控制解决方案：Telnet 服务、SSH 服务和 VNC 服务。

第 5 章主要介绍 NFS 和 NIS 的原理及其服务器端的配置方法，客户端的配置和测试等内容，最后结合 NFS 和 NIS 来实现对分布式系统用户账号和根目录的统一管理。

第 6 章主要介绍 DHCP 服务的工作原理，重点介绍 DHCP 服务器端动态 IP 地址分配和静态 IP 地址分配的配置方法。

第 7 章主要介绍 DNS 的层次式域名结构、DNS 查询的基本原理，Forwarding 服务器、根 DNS 服务器和主从服务器的配置方法。

第 8 章分别介绍 Apache 和 Tomcat 服务器的安装、配置和启动。并通过一个简易的 Web 工程讲述利用 MyEclipse 创建 Web 工程的基本步骤。

第 9 章主要介绍 FTP 的基本工作原理，以及使用 vsftpd 服务器软件架设 FTP 服务器的方法。

第 10 章主要介绍电子邮件服务的基本概念及工作原理，及两种不同的邮件服务器软件(Sendmail 和 Postfix)的安装和配置方法。

第 11 章主要介绍流行数据库服务器软件 MySQL 的安装、配置和使用，并介绍如何以图形化的方式来方便地管理 MySQL 服务器。

第 12 章主要介绍如何利用 OpenLDAP 软件架设目录服务器，涉及 OpenLDAP 软件的安装、配置和启动，以及如何使用 LDAP Browser/Editor 软件来管理 LDAP 服务器。

第 13 章主要介绍 Samba 服务的概念和原理、Samba 服务器端的配置和启动以及 Windows 系统和 Linux 系统 Samba 客户端的配置方法。

第 14 章主要介绍网络时间服务器的概念，着重介绍架设一台位于第三层的 NTP 服务器的配置方法，最后分别介绍 Linux 和 Windows 两种客户端的配置方法。

第 15 章主要介绍 Linux 图形化管理工具 Webmin 的安装和配置，以及利用 Webmin 管理图形化界面分别对 DHCP、Samba、DNS、Web、NFS、SSH、MySQL、防火墙等服务的配置方法，以及利用 Webmin 管理系统软件。

第 16 章主要介绍服务器性能监控的意义，以及国际上流行的服务器性能监控软件 Ganglia 的应用现状；然后介绍 Ganglia 的安装、启动和测试；并结合大型的应用案例，介绍网络服务器性能监控系统的开发步骤。

2．本书特色与优点

(1) 系统详细讲述各种服务器的架设技术

本书所讲述的 Linux 系统中服务器的架设和配置方法，其范围涵盖网络应用所涉及的大部分技术，包括 Linux 防火墙、远程控制、NFS 和 NIS、DHCP、Web 服务器、电子邮件、MySQL 数据库、LDAP、Samba、NTP 等，读者如能对这些内容全面掌握，相信定能轻松自如地架设各类 Linux 服务器。

(2) `理论性与实用性并重

本书在介绍各种 Linux 服务器架设和配置时，坚持理论与实用并重的原则。作者在多年的学习和研究中体会到只有对理论有清晰、深刻的理解，才能在实践中知其然并知其所以然，从而有的放矢地解决实践中所遇到的一些问题，因此，作者本着三分理论、七分实践的原则编写本书。

(3) 结合应用实例

很多服务器的架设只有在不同网段、多层次的实际局域网环境中才能显出其意义和配置的区别，因此，作者在讲述这些服务器的架设和配置时坚持以实际的网络拓扑为背景，以具体问题的求解为导向的原则，并且书中的一些实例将多种 Linux 服务器的配置方法和 Java/JSP、shell 编程、数据库技术等多种实用技术综合到了一起，旨在为读者解决大型网络应用问题提供思路和借鉴。

(4) 阐释前沿网络技术

微型计算机功能的日益强大和高速局域网技术的飞速发展加快了计算机系统由集中式走向分布式的进程，促进了以 P2P 计算和网格计算为代表的分布式计算技术的飞速发展。所谓分布式计算，就是数百台甚至上千台微机通过高速局域网连接在一起形成超级计算机以提供高性能的计算能力。分布式计算是目前活跃在网络最前沿的技术之一，为此，本书将简要介绍几种分布式计算的体系架构，并重点介绍 NFS、NIS、Web 服务器及网络服务器监控等技术在分布式系统中的应用。

3. 本书读者定位

本书面向广大工程技术人员及高等学校的教师及学生等，既可作为 Linux 服务器架设与管理方面的教材，又可作为各类培训班的培训教材。

本书由伍之昂、汤楠、庄毅主编，全书框架结构由何光明拟定。南京邮电大学的汤楠编写了第 1 章、第 2 章、第 11 章以及其他章节的概述部分；其他章节由伍之昂编写；浙江大学的庄毅博士以其丰富的计算机网络和数据库方面的学识，在全书的体系结构、理论阐释和实例选取等方面提出了许多精辟的见解。另外，张伍荣、吴婷、陈玉旺、许娟、吴蕾、姜萍萍、赵传申、杨明、杨萍、陈芳、范荣钢、钱阳勇、陈智、张凌云、王国全、丁善祥等同志对本书的出版也作出了重要的贡献，在此一并致谢。在本书的编写过程中，参考了许多相关书籍和资料，在此谨向这些参考文献作者表示深深的谢意。

由于作者水平有限，书中难免存在错漏和不妥之处，敬请广大读者和专家批评指正。

作 者

2008 年 3 月

目　　录

第 1 章　Linux 初阶

Linux 是一套免费开放的类 UNIX 操作系统，由于具有稳定可靠、高效灵活的特点使其成为架设网络服务器的主流操作系统。作为本书的开篇，本章将从 Linux 的起源开始，介绍 Linux 的优点、Fedora 与 Linux 的关系等内容；接着重点介绍 Fedora 6 的安装过程；最后介绍后续章节需要用到的 Linux 系统的一些基本知识，包括 vi 文本编辑器、文件和目录权限、shell 脚本的编写以及 Linux 系统用户管理等。

通过本章的学习，读者应掌握以下内容：

- ✧ Linux 的优势
- ✧ Fedora 与 Linux 的关系
- ✧ Fedora 6 的安装
- ✧ Linux 系统的一些基本操作

1.1　Linux 简介

1.1.1　Linux 的起源与优势

Linux 是一套免费使用和自由传播的类 UNIX 操作系统，它主要用于基于 Intel x86 系列 CPU 的计算机上。这个系统是由世界各地成千上万的程序员设计实现的，其目的是成为不受任何商品化软件的版权制约的、全世界都能自由使用的 UNIX 兼容产品。

Linux 的出现，最早开始于一位名叫 Linus Torvalds 的计算机爱好者，当时他是芬兰赫尔辛基大学的学生。他的目的是想设计一个代替 Minix 的操作系统(Minix 是世界著名教授 Andrew Tannebaum 编写的操作系统教程上的一个实例操作系统)，这样才有了 Linux 的雏形。

Linux 以其高效性和灵活性著称。它能够在 PC 上实现全部的 UNIX 特性，具有多任务、多用户的能力。Linux 可以在 GNU 公共许可权限下免费获得，是一个符合 POSIX 标准的操作系统。Linux 操作系统软件包不仅包括完整的 Linux 操作系统，而且还包括了文本编辑器、高级语言编译器等应用软件。它还包括带有多个窗口管理器的 X-Windows 图形用户界面，从而让用户像使用 Windows NT 一样，可以使用窗口、图标和菜单对系统进行操作。

Linux 之所以受到广大计算机爱好者的喜爱，主要原因有两个：一是它属于免费软件，用户不用支付任何费用就可以获得其源代码，并且可以根据自己的需要对它进行必要的修改，无偿进行使用，无约束地进行传播；另一个原因是，它具有 UNIX 的全部功能，任何使用 UNIX 操作系统或想学习 UNIX 操作系统的人都可以从 Linux 中获益。

在一贯由微软的 Windows NT、Novell 的 Netware 和 UNIX 占据的网络操作系统市场上，Linux 正日益成为一个令人生畏的对手。Linux 已摆脱了其最初仅限于 Linux 爱好者和研究机构所使用的业余软件的身份，更多地受到企业用户的重视。这一方面得益于其开放源码的措施，通过 Internet 上成千上万爱好者和开发者的不懈努力，Linux 比以往任何时候都更健壮、更稳定、更可靠；另一方面，则得益于众多像 Red Hat 这样的商业软件公司积极介入 Linux 产品及其服务领域，这极大地加快了 Linux 的商品化步伐，从而使企业用户可以更放心、更有保障地布置他们的 Linux 系统。随着以 IBM、Intel、Oracle、CA 及网景为代表的众多 IT 巨头们纷纷宣布支持 Linux，那些曾经对 Linux 不屑一顾的著名软件公司的老板们不得不回过头来重新审视这个由 Internet 上的一群业余人员所开发出来的操作系统。

那么 Linux 到底具备哪些优点呢？我们可以总结出以下几点。

- 多平台。虽然 Linux 主要在 x86 平台上运行，但是目前已经移植到下列平台：Alpha 和 Sparc。Red Hat 公司已经推出了这两个平台的发行套件，并且对 x86 平台支持多种 CPU，包括：Intel/AMD/Cyrix 的 386/486/Pentium 系列、Pentium Ⅱ、K6/M2/Cyrix 6x86 等。
- 对应用程序使用的内存进行保护。在 Linux 下，应用软件无法访问系统分配的内存以外的内存区域。这样，一个软件的错误操作不会造成整个系统的瘫痪。
- 按需取盘。在 Linux 下，任何一个执行文件在执行时，只有那些确实被用到的代码段才会被系统读取到内存中，这样节约了大量的读取磁盘时间，自然也就加快了程序执行速度，并且这是在操作系统级实现的，而不像 DOS 下是要靠应用程序 smartdrive 来管理，两者性能的差别是很大的。
- 共享内存页面。在 Linux 下，多个进程可以使用同一块内存页面(每块大小为 4KB)，只有当某一个进程试图对这块页面执行写操作时，Linux 才把这块页面为该进程复制到内存的另一块区域(copy-on-write)，这样做的好处是不仅加快了程序运行的时间，还节约了宝贵的物理内存。
- 优秀的磁盘缓冲调度功能。Linux 最突出的优点就是它的磁盘 I/O 速度，因为它将系统没有用到的剩余物理内存全部用来作为硬盘的高速缓冲，当有对内存要求比较大的应用程序运行时，它会自动地将这部分内存释放出来给应用程序使用。这比 DOS/Windows 下的 smartdrive 只能呆板地使用固定大小的缓冲区要先进得多。
- 动态链接共享库。同 Windows 95 的 DLL 一样，Linux 也使用动态链接共享库(同时当然也提供静态链接库)。这个特性可以大大减小 Linux 应用程序的大小，并且被很多程序同时调用的一段代码只要被加载一次就可由众多程序共享。
- 支持多种文件系统。Linux 支持的文件系统的种类包括 minix、ext、ext2、xiafs、hpfs、fat、msdos、umsdos、vfat、proc、nfs、iso9660、smbfs、ncpfs、affs、ufs、romfs、sysv、xenix 和 cohernet。Linux 可以将这些文件系统直接装载(mount)为系统的一个目录。Linux 自己的文件系统 ext2fs 是非常先进的，最多可以支持到容量为 2TB 的硬盘，文件名长度的限制为 255 个字符。同时 Linux 还支持以只读方式打开 HPFS-2 格式的 OS/2 2.1 文件系统和 HFS 格式的 Macintosh 文件系统。
- 强大的网络功能。Linux 支持所有常见的网络服务，包括 FTP Telnet、NFS 等。Linux 在最新发展的核心中包含的基本协议有 TCP、IPv4、IPv6、AX.25、X.25、IPX、

DDP(Appletalk)、NetBEUI 和 Netrom 等。目前，稳定的核心中包含的网络协议有 TCP、IPv4、IPX、DDP 和 AX 等，另外还提供 Netware 的客户机和服务器，以及现在最热门的 Samba(让用户共享 Mircosoft Network 资源)。Linux 还包括 Appletalk 服务器。

其他的特点还包括：支持 POSIX 的任务控制；软件移植性好；与其他 UNIX 系统有良好的兼容性等。

基于 Linux 的显著优势，它在服务器操作系统中一定会变得越来越流行，因此如何在 Linux 系统下架设各类服务器将成为网络管理员的必修课。

1.1.2　Fedora 简介

Linux 自诞生以来，像其他许多软件一样发布了很多不同的版本，最常见的有 Slackware、RedHat、Debian、S.u.s.E.等。Fedora Core(有时又称为 Fedora Linux)是众多 Linux 发行版本之一，它是一套从 Red Hat Linux 发展出来的免费 Linux 系统。Fedora 和 Red Hat 这两个 Linux 的发行版本联系很密切。Red Hat 自 9.0 以后，不再发布桌面版，而是把这个项目与开源社区合作，于是就有了 Fedora 发行版。Fedora 可以说是 Red Hat 桌面版本的延续，只不过是与开源社区合作。

Fedora 是一个开放的、创新的、前瞻性的 Linux 操作系统和平台，它允许任何人自由地使用、修改和重发布，无论现在还是将来。它由一个强大的社群开发，这个社群的成员以自己的不懈努力，提供并维护自由、开放源码的软件和开放的标准。Fedora 项目由 Fedora 基金会管理和控制，得到了 Red Hat Inc.的支持。Fedora 项目的目标是与 Linux 社区一同构造一个完整的、通用的操作系统。Fedora 项目计划每隔一定的时间发布 Fedora Core，大约每年 2～3 次，可以查看公开的日程表。Red Hat 工程师团队一直参与构建 Fedora Core 的过程中，同时邀请并鼓励更多其他人参与其中。通过使用这种开放的过程，他们希望可以提供一个更加贴近自由软件理想，同时更受开源社区欢迎的操作系统。

Fedora Core 被红帽公司定位为新技术的实验场，与 Red Hat Enterprise Linux 被定位为稳定性优先不同，许多新的技术都会在 Fedora Core 中检验，如果稳定的话红帽公司才会考虑加入到 Red Hat Enterprise Linux 中。到目前为止，Fedora Core 已经发行了 7 个版本，最新版本为 Fedora Core 7。其发行历史如下。

2003 年 11 月，第一个发行版本 Fedora Core 1 出炉，版本代码为 Yarrow。这一版本与 Red Hat Linux 非常相似，除加入了新的安装机制 yum 以外，只是把 Red Hat 的标志换掉，并更新套件而已。

2004 年 5 月，Fedora Core 2 正式发布，版本代码为 Tettnang。这一版本除了是第一个采用 2.6.9 版核心的发行套件及用 Xorg X11 取代 XFree86 外，也加入了 IIIMF、SELinux 等许多新技术，并且在开放源代码社群的支持下修正了许多套件的错误。

2004 年 11 月，Fedora Core 3 正式发布，版本代码为 Heidelberg。这一版本采用 2.6.9 版核心、Xorg 6.8.1、GNOME 2.8 和 KDE 3.3.0。

2005 年 6 月，Fedora Core 4 正式发布，版本代码为 Stentz。这一版本采用 2.6.11 版核心、GNOME 2.10、KDE 3.5、GCC 4.0 和 PHP 5.0。此外，FC4 还增加了对 PowerPC 的支持。

2006 年 3 月 20 日，Fedora Core 5 已经正式发布，版本代码为 Bordeaux。GNOME 桌面基于 2.14 发布，KDE 桌面是 3.5 的一般版本。这一版本首次包含 Mono 支持，以及众多 Mono 应用程序，例如 Beagle 桌面搜索工具、F-Spot 相片管理工具以及 Tomboy 记事程序。SCIM 语言输入框架取代了过去使用的 IIIMF 系统。默认网页浏览器是 Firefox 1.5，gcc 4.1 编译器包含其中，内核基于 Linux 2.6.15。

2006 年 10 月 24 日，新版本 Fedora Core 6 于美国东部时间上午 10 时正式发布。

2007 年 6 月 2 日，最新版本的 Fedora 7 正式发布。

由于成书时间所限，本书选用的版本是 Fedora Core 6(简称 Fedora 6)，书中所讲述的所有服务器的架设和配置实验都是在 Fedora 6 系统下测试通过。

1.2 Fedora 6 的安装

Fedora 6 安装盘如果是 VCD 格式的，则包含 6 张，也可能是 1 张 DVD 格式的安装盘。下面将介绍 Fedora 6 的安装过程。

1. 插入 Fedora 6 的安装盘后，进入机器 BIOS，设置为光盘启动，如图 1-1 所示。机器通过光盘启动后，弹出如图 1-2 所示的界面，按 Enter 键选择图形化的方式安装，弹出如图 1-3 所示的界面，说明机器已经进入正常的 Fedora 安装程序。

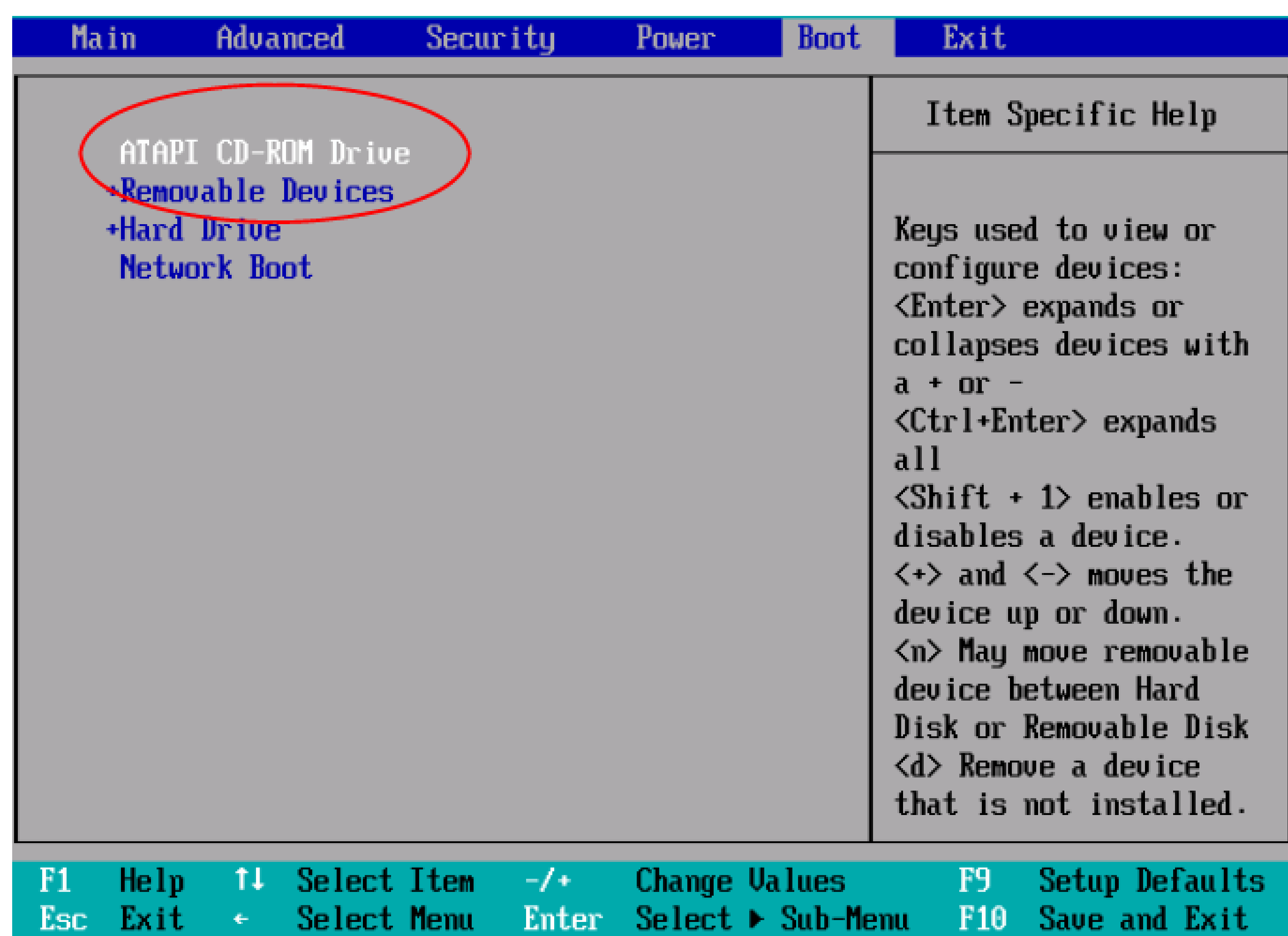


图 1-1　设置光盘启动



图 1-2　Fedora 6 开始安装的界面

```
Dentry cache hash table entries: 32768 (order: 5, 131072 bytes)
Inode-cache hash table entries: 16384 (order: 4, 65536 bytes)
Memory: 249064k/262144k available (2105k kernel code, 12412k reserved, 844k data
, 240k init, 0k highmem)
Checking if this processor honours the WP bit even in supervisor mode... Ok.
Calibrating delay using timer specific routine.. 2604.53 BogoMIPS (lpj=5209064)
Security Framework v1.0.0 initialized
SELinux:  Initializing.
SELinux:  Starting in permissive mode
selinux_register_security:  Registering secondary module capability
Capability LSM initialized as secondary
Mount-cache hash table entries: 512
CPU: L1 I Cache: 64K (64 bytes/line), D cache 64K (64 bytes/line)
CPU: L2 Cache: 256K (64 bytes/line)
Intel machine check architecture supported.
Intel machine check reporting enabled on CPU#0.
Checking 'hlt' instruction... OK.
SMP alternatives: switching to UP code
Freeing SMP alternatives: 12k freed
ACPI: Core revision 20060707
CPU0: AMD Athlon(TM) XP 2100+ stepping 02
Total of 1 processors activated (2604.53 BogoMIPS).
ENABLING IO-APIC IRQs
..TIMER: vector=0x31 apic1=0 pin1=2 apic2=-1 pin2=-1
_
```

图 1-3　按 Enter 键启动系统

❷ 进入 Fedora 6 的安装程序后，安装程序会询问是否检验光盘，初始界面如图 1-4 所示。如果无需检验光盘可以单击 Skip 按钮跳过此步；如果单击 OK 按钮则出现如图 1-5 所示的界面，提示插入任意一张 Fedora 6 进行测试，插入第 1 张光盘后，单击 Test 按钮进行光盘测试，过程如图 1-6 所示。测试成功后，弹出测试结果窗口，如图 1-7 所示，结果显示第 1 张光盘测试成功，可以依照同样的步骤测试其他光盘。



图 1-4　询问是否检验光盘

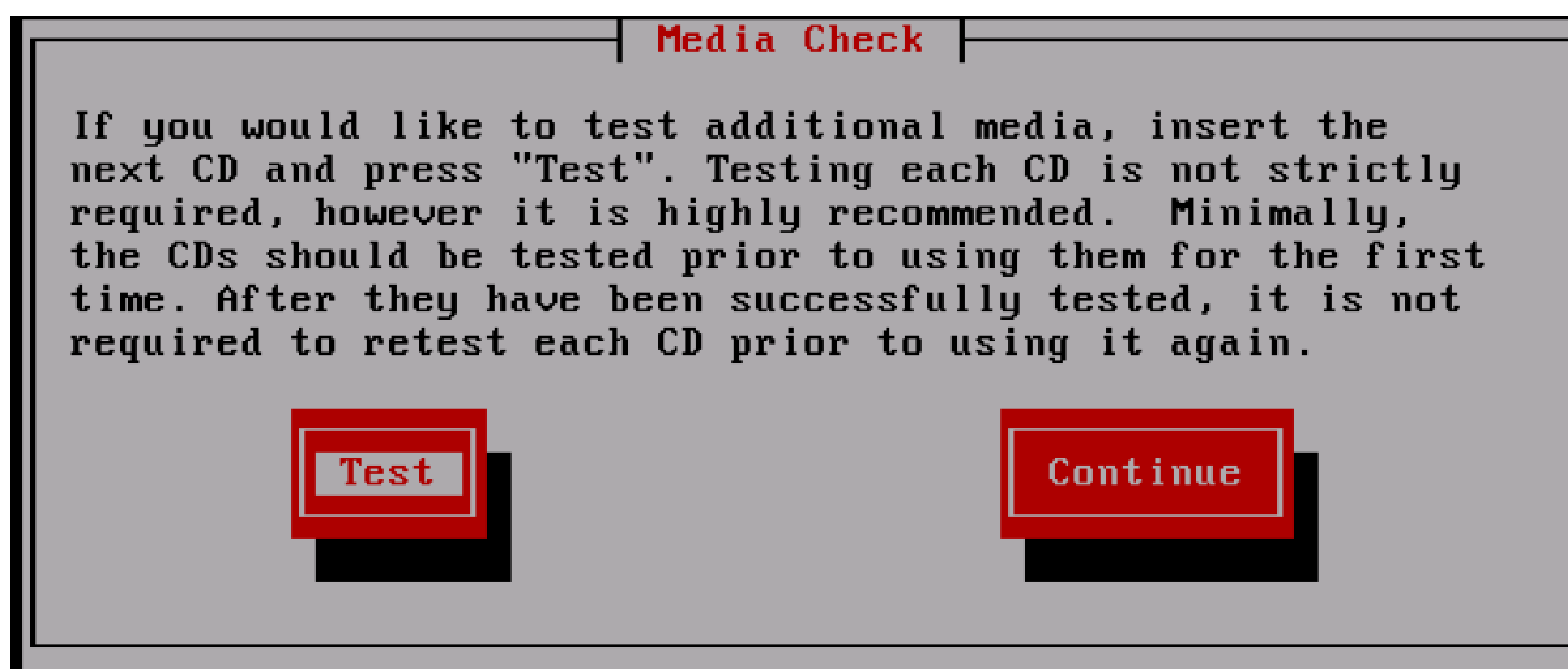


图 1-5　插入光盘进行检验

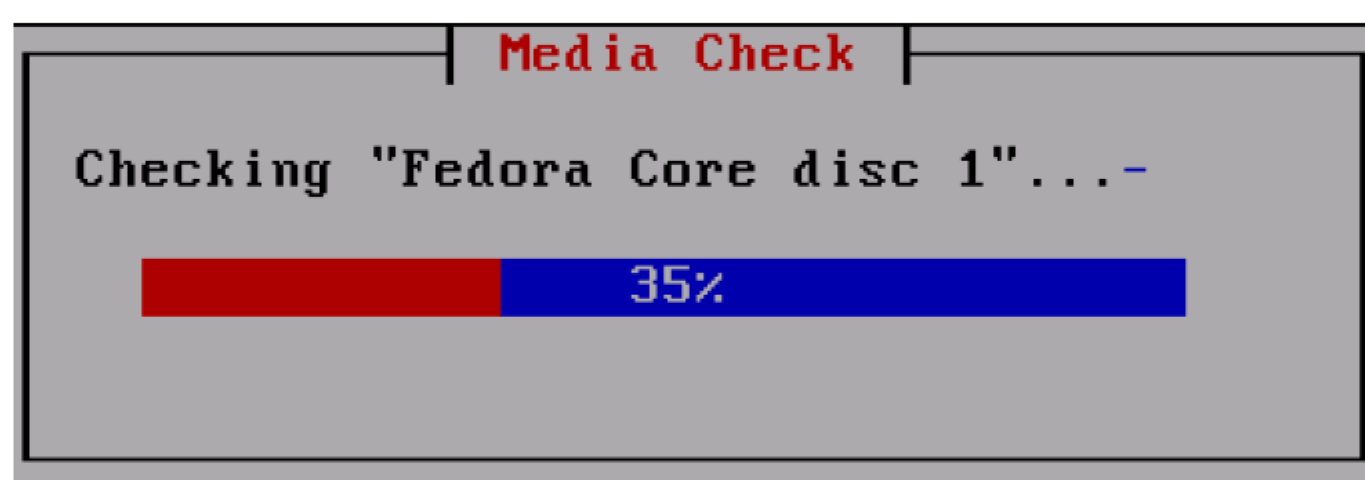


图 1-6　检验光盘过程



图 1-7　光盘检验结果

❸ 完成检验光盘后，弹出如图 1-8 所示的界面。单击 Next 按钮进入选择语言界面，Fedora 6 支持多国语言，可以根据需要选择【简体中文】、【繁体中文】或 English，如图 1-9 所示。单击 Next 按钮进入选择键盘的界面，这里一般选择【美国英语式】选项，如图 1-10 所示。

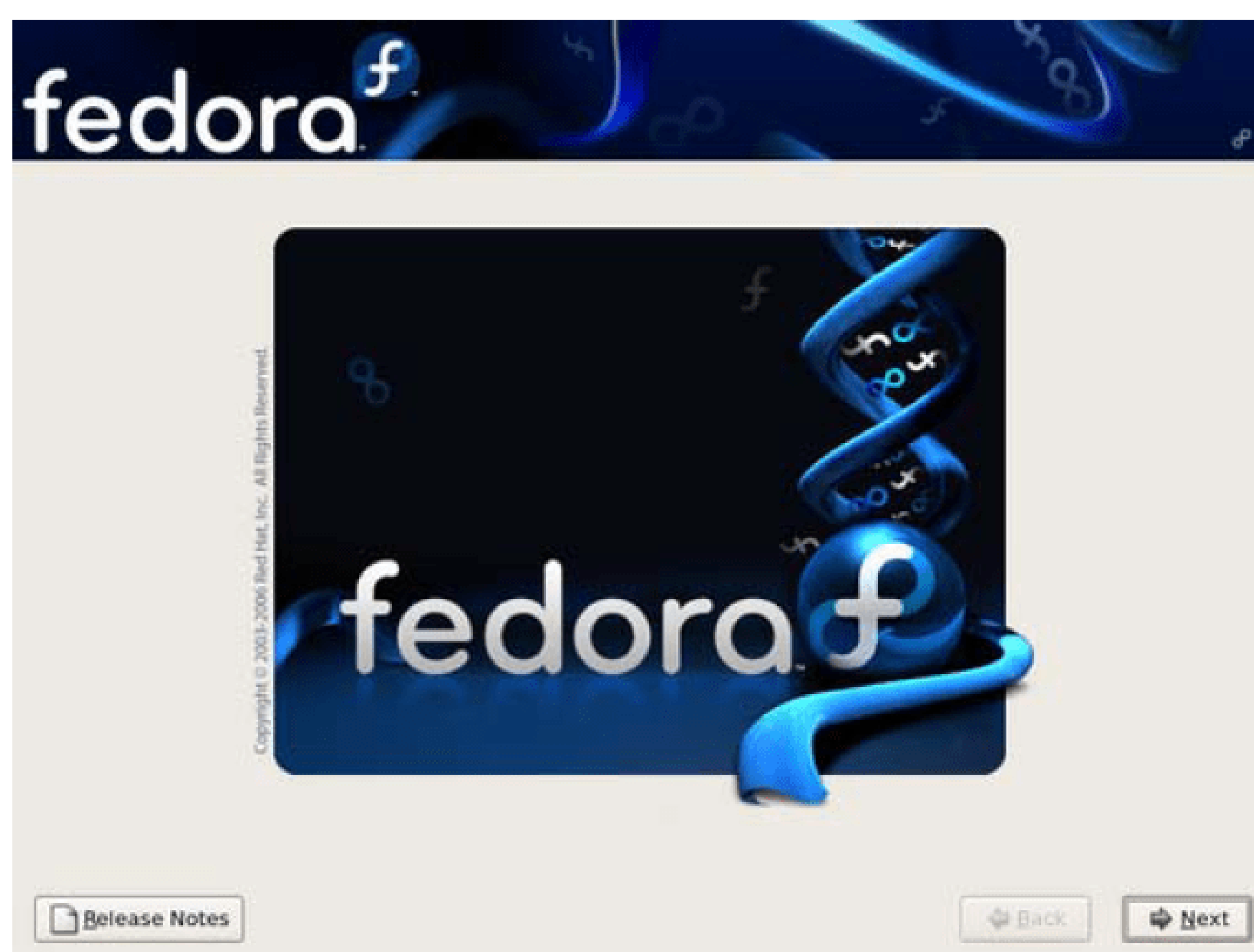


图 1-8　图形化安装第一步

❹ 如果服务器上已经安装过 Fedora，那么单击 Next 按钮将出现如图 1-11 所示的界面，提示选择需要重新安装还是升级系统。第一次安装 Fedora 则直接进入下一步的操作。

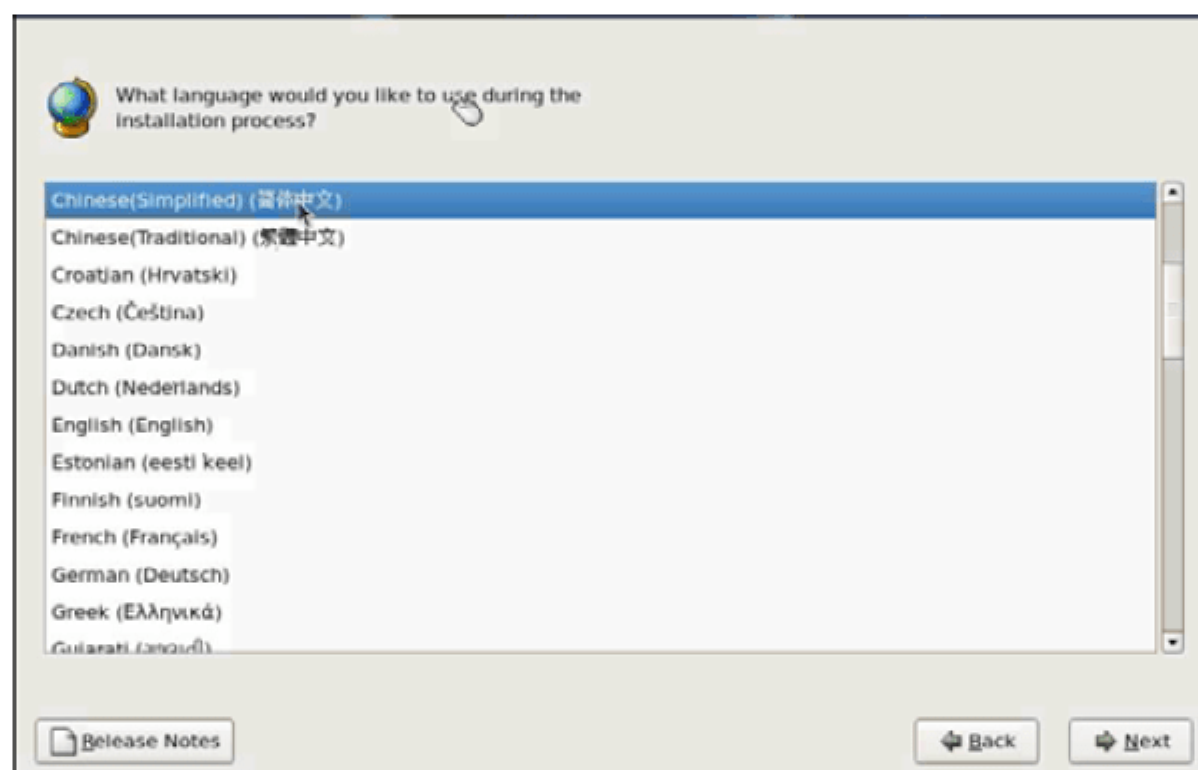


图 1-9　选择系统语言

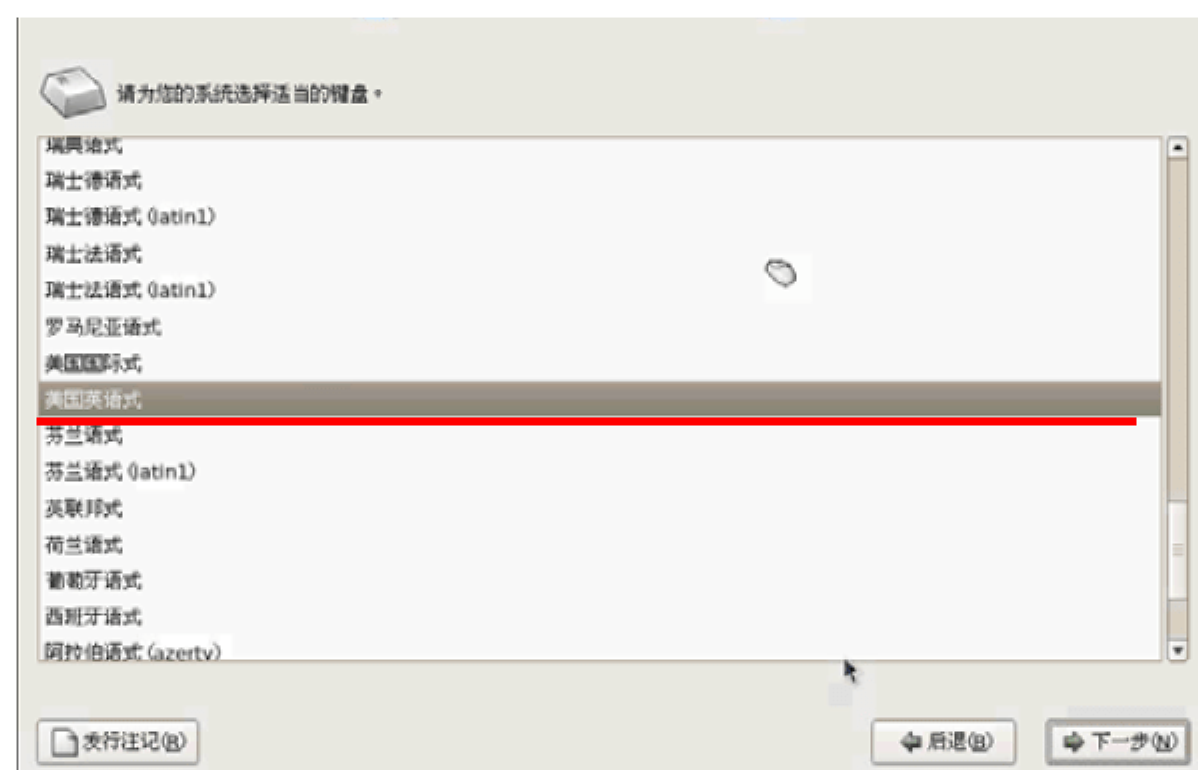


图 1-10　选择键盘种类

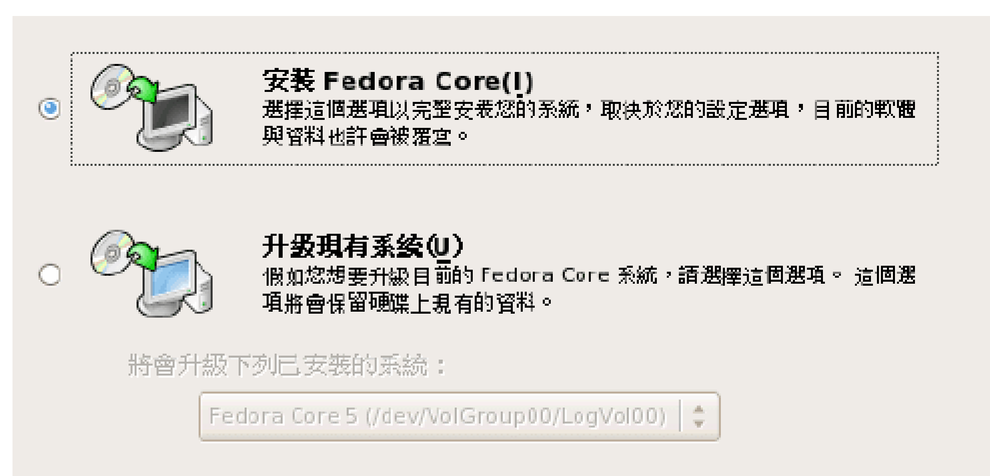


图 1-11　选择安装或升级

❺ 单击 Next 按钮弹出如图 1-12 所示的界面，提示将要进行磁盘分区操作，该操作将格式化硬盘数据，需要用户确认。单击【是】按钮进入如图 1-13 所示的磁盘分区界面，在图中标注的微调框中选择【建立自定义的分区结构】选项。

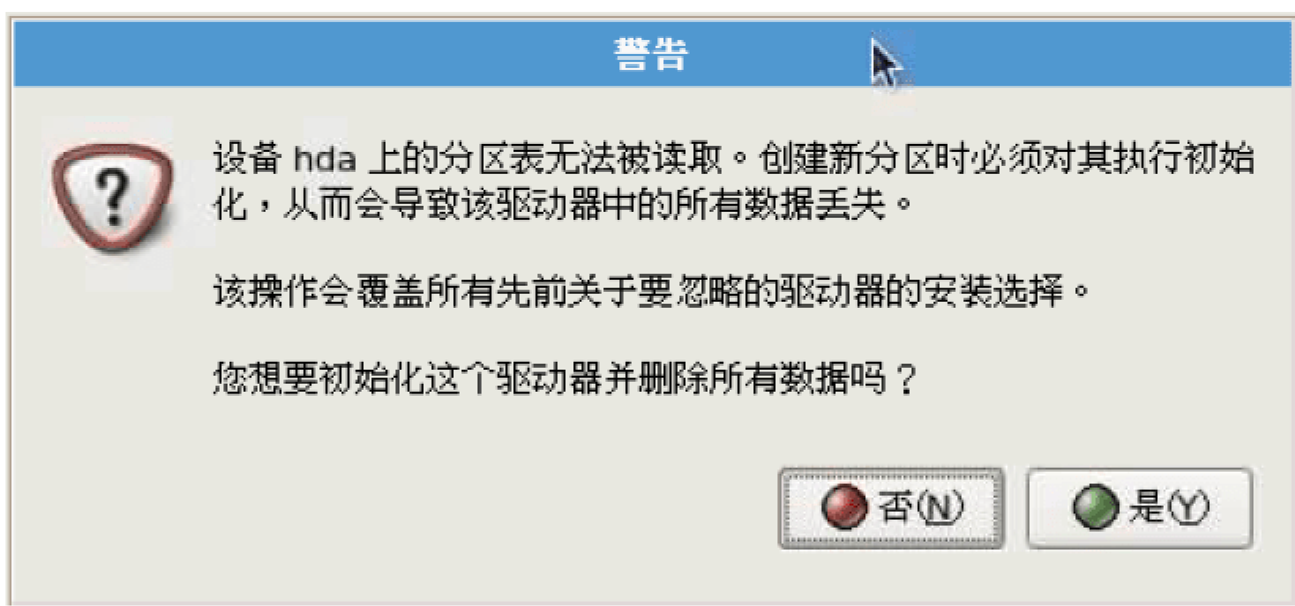


图 1-12　确认格式化磁盘操作

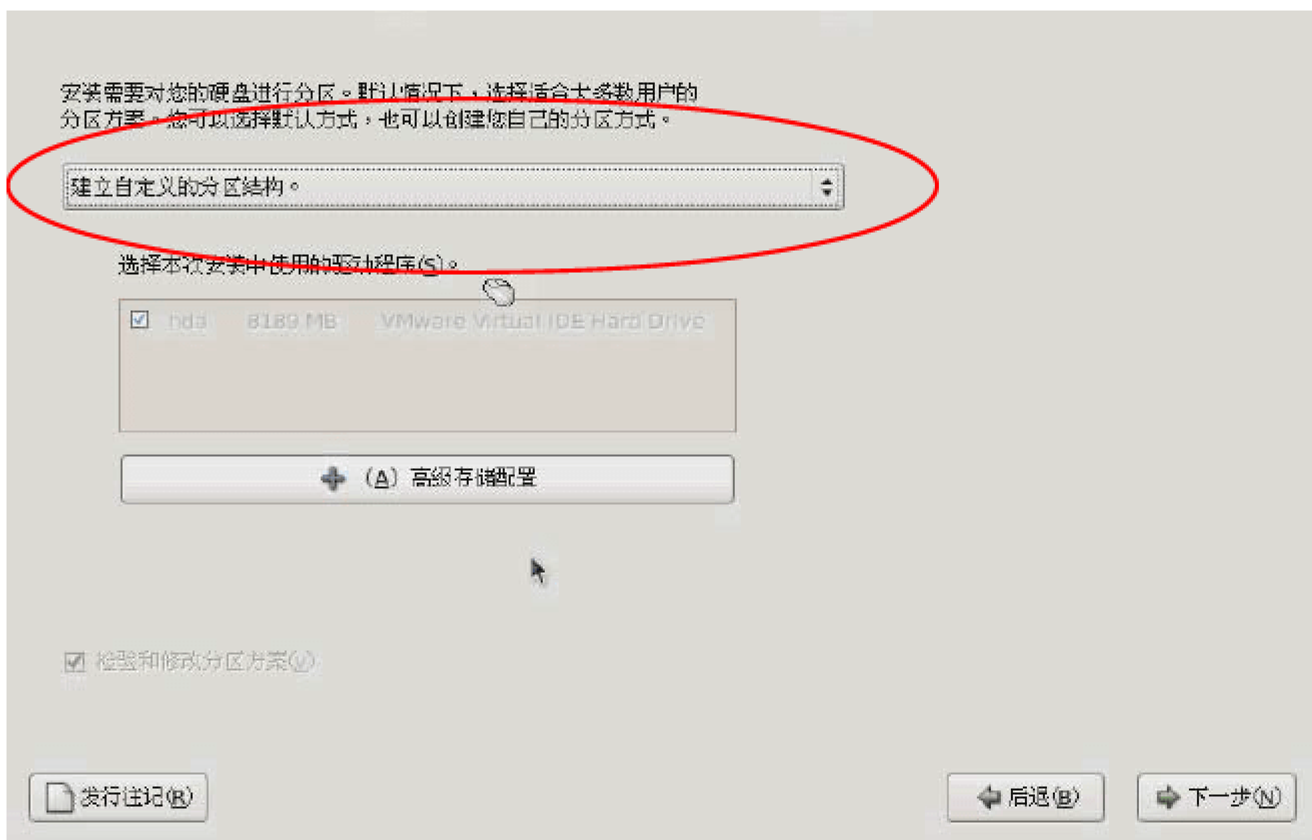


图 1-13　选择磁盘分区方式

❻ 单击【下一步】按钮进入 Linux 的 Disk Druid 程序，如图 1-14 所示。Disk Druid 程序类似于 Partition Magic 程序，都是用于磁盘分区，不过 Disk Druid 程序划分的是 Linux 承认的分区，与 Windows 下的 FAT 或 NTFS 一样属于磁盘的不同格式。该界面分为三部分。

① 顶部的长条窗口显示了已经划分的硬盘的信息。如果该机器有两个或两个以上的硬盘，此处就显示多个长条窗口。

② 中间的按钮区提供新建分区、删除分区和编辑分区等操作。

③ 底部的窗口显示不同分区的信息，包括硬盘名称、挂载点(Mount Point)、文件系统类型、是否进行格式化及分区大小等。

Linux 系统要求最少有两个分区，分别为：

① ext3 存放文件的分区，即文件系统的存放区域；

② swap 用作虚拟内存(Virtual Memory) 的分区。

由于 ext3 分区存放了 Linux 的文件系统，因此该分区应该尽量大些，建议除了 swap 分区以外的磁盘都分给 ext3 分区。

下面介绍如何创建 ext3 和 swap 分区，首先是 ext3 分区，选定剩余空间，单击【新建】按钮，弹出如图 1-15 所示的窗口，其中的【大小】微调框设定了分区的大小，以兆字节(MB)

为单位，对于 ext3 分区不改动这里，而以下面额外的大小选项决定分区的大小，在下面的【其它大小选项】选项组中选中【使用全部可用空间】单选按钮，如图 1-15 所示。

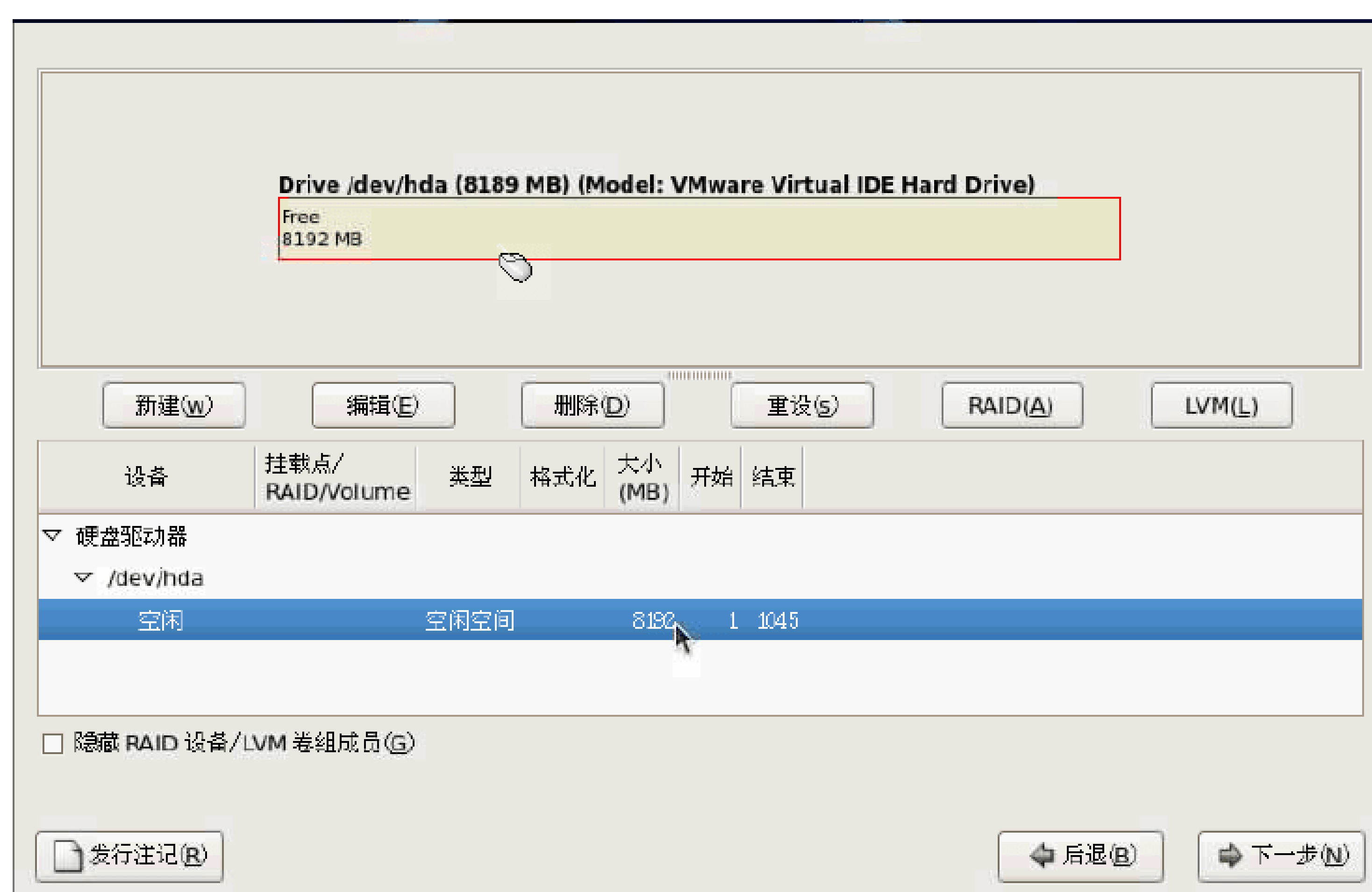


图 1-14　Disk Druid 磁盘分区程序主界面

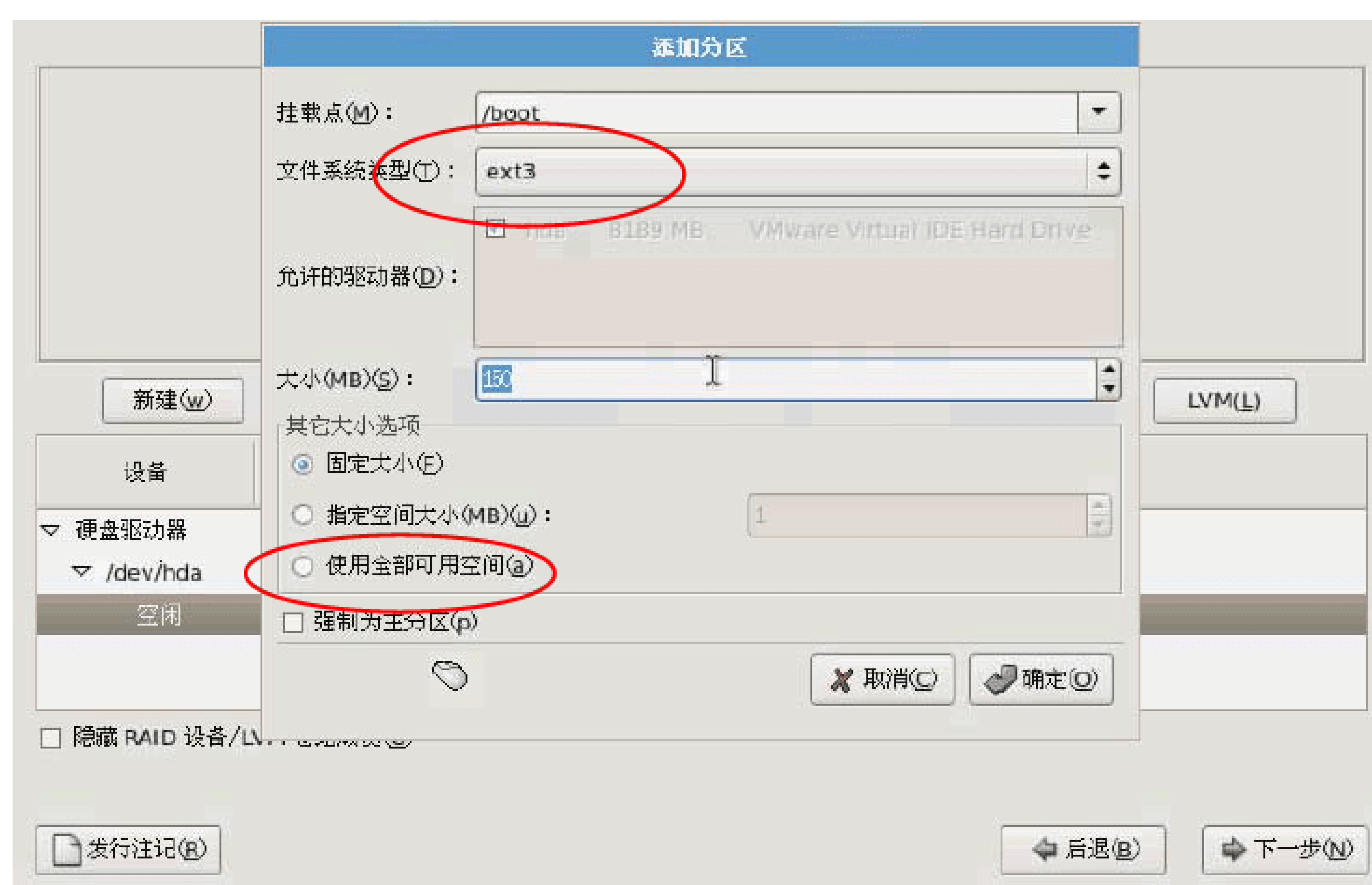


图 1-15　添加 ext3 分区

接着创建 swap 分区，同样选定剩余空间，单击【新建】按钮，弹出如图 1-16 所示的窗口，选择【文件系统类型】为 swap，在【其它大小选项】选项组中选中【固定大小】单选按钮，将 swap 大小设置为 512MB。

创建 ext3 和 swap 分区成功后，弹出如图 1-17 所示的窗口，它显示了当前的分区和磁盘信息。

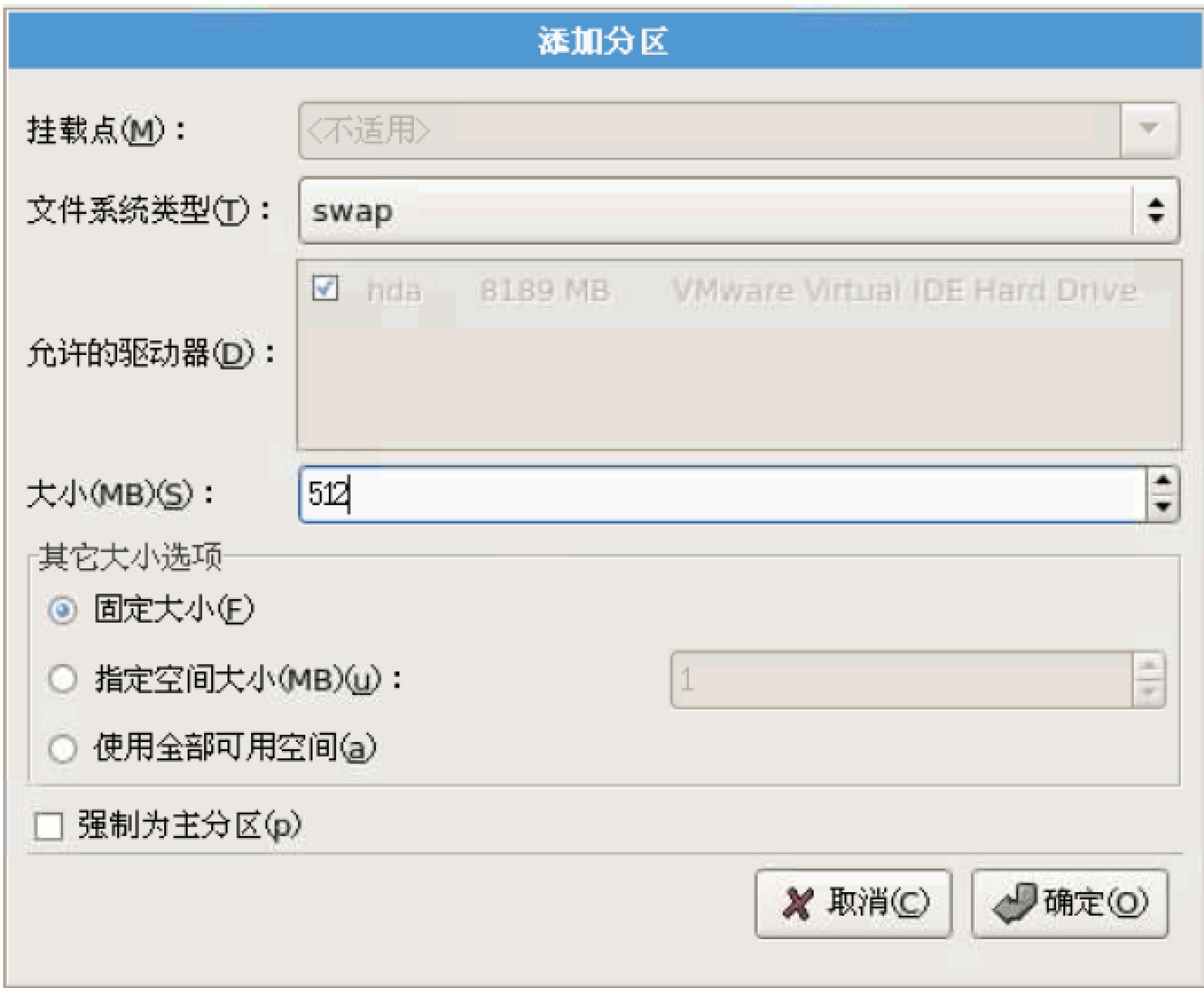


图 1-16　添加 swap 分区



图 1-17　添加分区后的结果

❼ 接下来是网络设置，在此可以不做设置，直接忽略此步，等 Linux 系统安装完毕后再设置网络，具体设置将在 2.2 节“Linux 系统的网络配置”中详细讲述。

跳过网络设置，进入系统时区设置步骤，可选择亚洲/上海时区，如图 1-18 标注部分所示。



图 1-18　设定所在时区

❽ 接下来是设定系统管理员密码，即 root 用户密码，如图 1-19 标注部分所示。

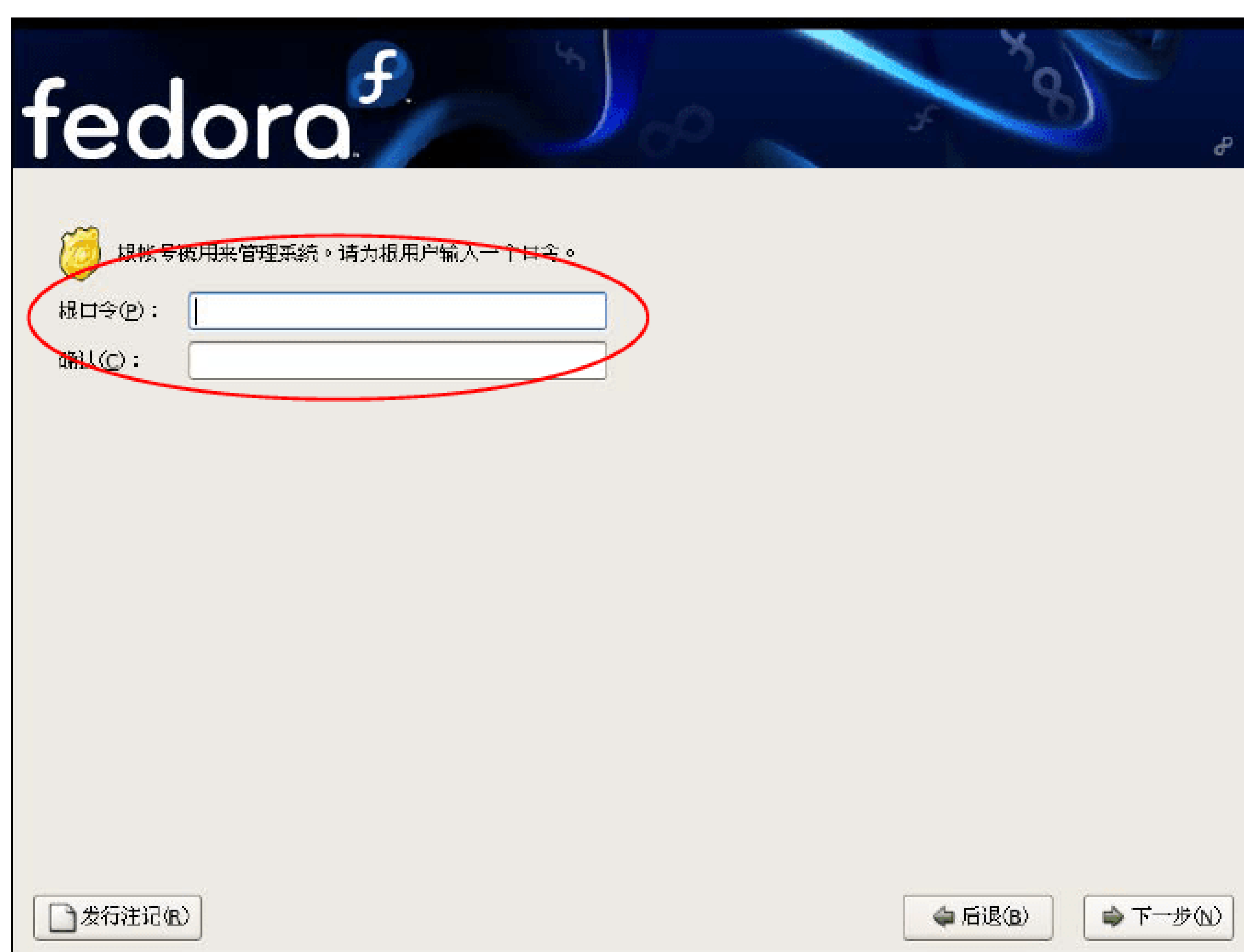


图 1-19　设定管理员密码

❾ 设定系统管理员密码完毕后，可选择所安装软件步骤，如图 1-20 标注部分所示，选择不同的额外功能模块，Fedora 将安装不同的第三方软件，当然也可以根据用户需要选中【现在定制】单选按钮，进行软件的定制。

然后单击【下一步】按钮，在出现的界面中会提示 Fedora 设置完毕，将进入安装 Fedora Core 的过程，如图 1-21 所示。



图 1-20　选择安装软件窗口



图 1-21　Fedora 安装设置完毕

⑩ 单击【下一步】按钮，进入 Fedora 6 的安装过程，如图 1-22 所示。安装的时间取决于机器的性能，如果用户使用的是 6 张 VCD 的 Fedora 6 安装盘，系统会提示用户更换光盘，图 1-23 展示了 Fedora 6 的安装过程，安装完毕后弹出如图 1-24 所示的窗口，该窗口提示将安装盘从光驱中取出，然后通过单击【重新引导】按钮来重启机器。



图 1-22　开始安装 Fedora 6



图 1-23　Fedora 6 安装过程

⓫ 重启计算机后，首先出现如图 1-25 所示的 Boot 界面，然后出现如图 1-26 所示的等待系统启动界面。第一次启动 Fedora 系统时，还会出现如图 1-27 所示的欢迎界面，根据提示单击【前进】按钮同意委托授权书，并完成设置防火墙、设置系统时间等操作。



图 1-24　安装完成



图 1-25　Fedora Boot 界面

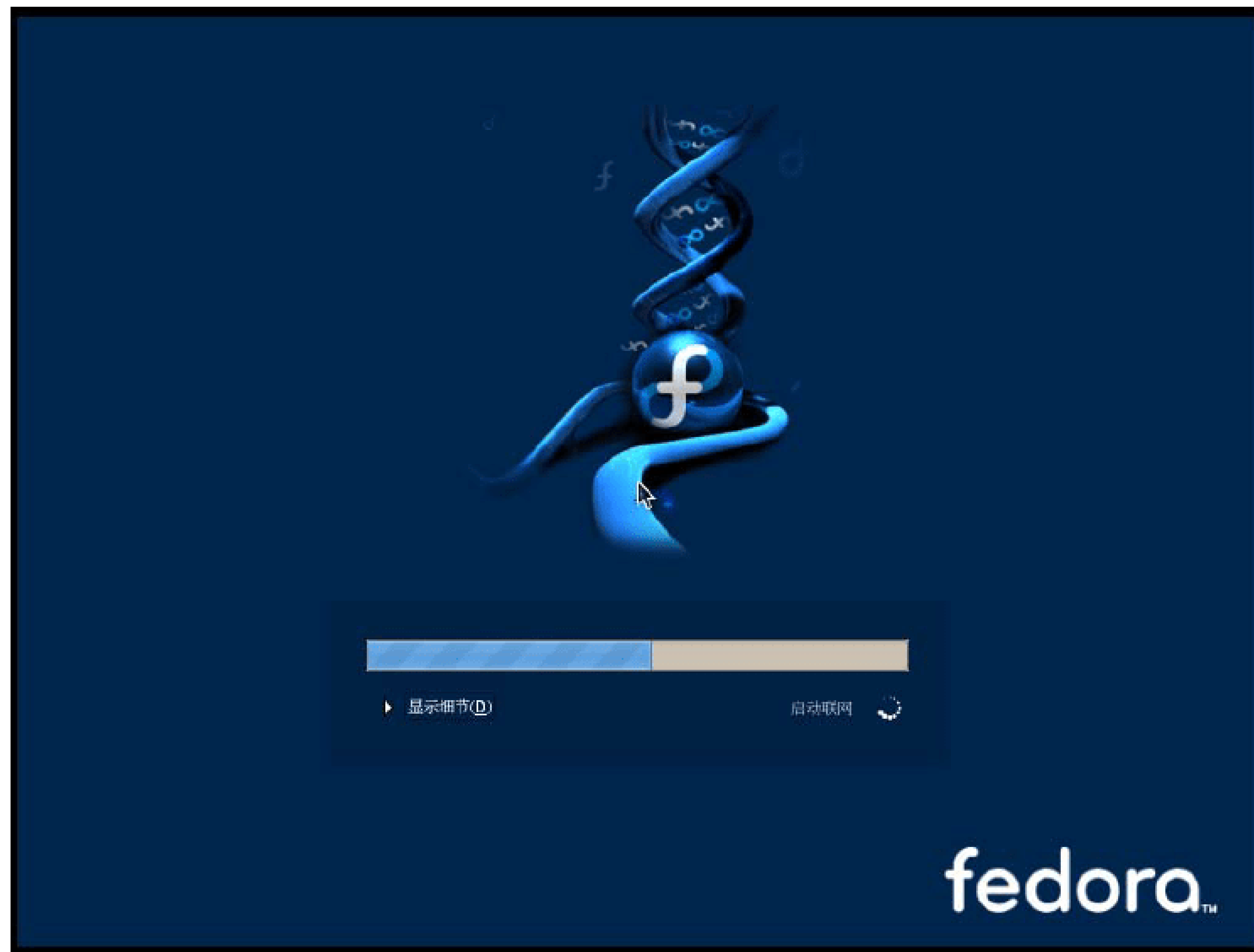


图 1-26　进入系统



图 1-27　Fedora 的欢迎界面

⑫ 设置欢迎界面完毕后，进入 Fedora X-Win 的登录界面，如图 1-28 所示。在此输入 root 用户或者其他用户的名称和密码，就可以进入 X-Win 的窗口界面，如图 1-29 所示。X-Win 类似于 Windows 桌面窗口，桌面上是文件系统、回收站等图标；窗口上方是开始菜单，从中可以选择更多的系统程序。



图 1-28　Fedora 登录界面



图 1-29　X-Win 窗口界面

📖**点评与拓展：**磁盘分区是 Fedora 6 安装过程中最关键的步骤，建议 swap 分区的大小为物理内存(Physical Memory)大小的两倍。如果计算机需要进行大量的软件编译工作，swap 分区应设置得稍微大些，但 swap 分区过大将会浪费系统资源，对于初学者来说，两倍定律比较适合。

1.3 预 备 知 识

1.3.1 vi 文本编辑器

Linux 系统最常用的文本编辑工具就是 vi 文本编辑器，它以命令行的方式处理文本，尽管不如图形化处理方式直观，但它以操作速度快、功能全面等优点赢得了 Linux 用户广泛的认可。Vi 可分为三种模式：一般模式、编辑模式和命令模式，下面分别对这三种模式的作用作简单介绍。

- ✧ 一般模式：输入 vi 命令进入 vi 文本编辑器的时候，就是一般模式了，这个模式允许用户移动光标查看文本内容及进行复制、粘贴操作；最有用的是它所提供的搜索功能，对于很长的文本文件，很多时候都需要搜索文本关键字进行快速定位，这个操作就在一般模式下进行。
- ✧ 编辑模式：从一般模式按下 i、o、a、r 等字母键进入编辑模式，在此模式下可以对文本进行编辑，按 Esc 键则从编辑模式退回到一般模式。
- ✧ 命令模式：从一般模式按下:、/、?等键进入命令模式，光标自动移到最底下一行，读取、存储、离开 vi 等操作均在此实现。

下面可通过一个简单的例子进一步说明 vi 文本编辑器的三种模式及其基本操作。

输入下面的命令打开或新建文本文件，如图 1-30 所示，进入一般模式，如图 1-31 所示。按 i 字母键进入编辑模式，窗口左下角出现 INSERT 或 REPLACE 字样表明进入了编辑模式，如图 1-32 标注部分所示。

```
[root@sec ~]# vi testvi
```

图 1-30 vi 命令



图 1-31 vi 的一般模式



图 1-32 vi 的编辑模式

文本编辑完后，按:键进入命令模式，输入 wq 表示存储并退出 vi，如图 1-33 所示，表 1-1 列出了 vi 命令模式的常用命令。

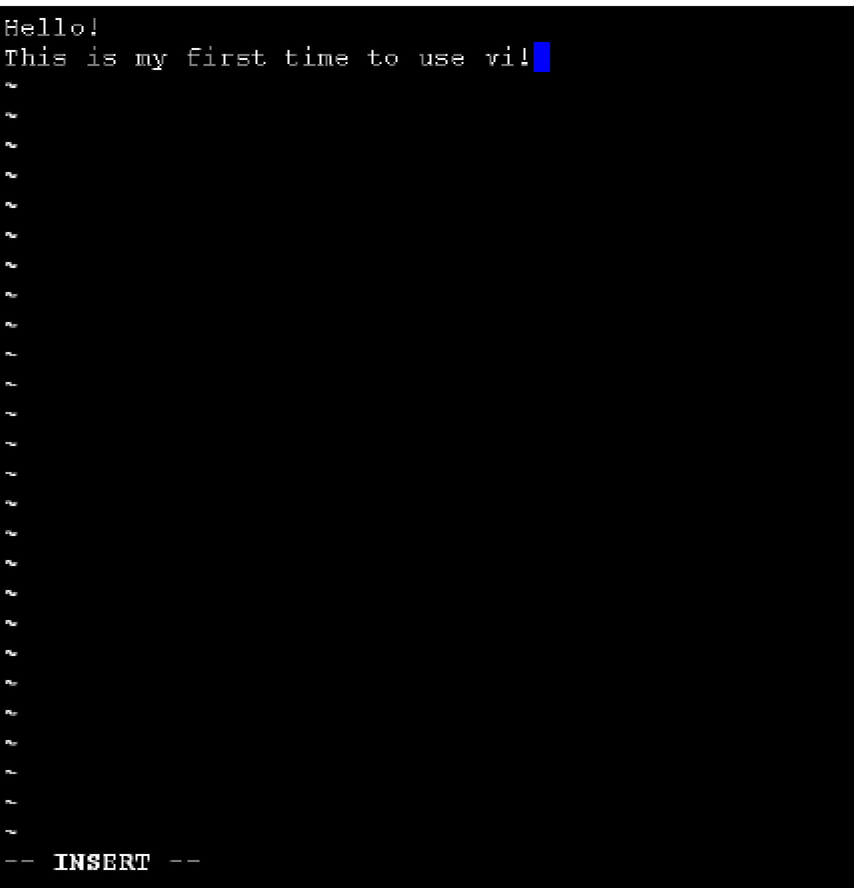


图 1-33 vi 的命令模式

表 1-1 vi 命令模式的常用命令

vi 命 令	描 述
:w	将编辑的文本存储
:q	离开 vi 文本编辑器
:q!	曾修改过文本，但是不想存储，使用该命令强制离开 vi 文本编辑器
:wq	存储文本并离开 vi 文本编辑器

除了文本编辑功能，文本搜索功能也是常用的功能，它可以实现长文本文件中的字符串快速定位。打开文本文件，在一般模式按/键并在窗口左下角输入搜索内容，如图 1-34 标

注部分所示，图中输入的是 HTTP，按 Enter 键就能搜索到文本中第一次出现 HTTP 字符串的地方，表 1-2 列出了 vi 一般模式的常用命令。

```
<!-- A "Server" is a singleton element that represents the entire JVM,
     which may contain one or more "Service" instances.  The Server
     listens for a shutdown command on the indicated port.

     Note:  A "Server" is not itself a "Container", so you may not
     define subcomponents such as "Valves" or "Loggers" at this level.
 -->

<Server port="8005" shutdown="SHUTDOWN" debug="0">


  <!-- Comment these entries out to disable JMX MBeans support -->
  <!-- You may also configure custom components (e.g. Valves/Realms) by
       including your own mbean-descriptor file(s), and setting the
       "descriptors" attribute to point to a ';' seperated list of paths
       (in the ClassLoader sense) of files to add to the default list.
       e.g. descriptors="/com/myfirm/mypackage/mbean-descriptor.xml"
  -->
  <Listener className="org.apache.catalina.mbeans.ServerLifecycleListener"
/HTTP
```

图 1-34　vi 搜索命令

表 1-2　vi 一般模式的常用命令

vi 命 令	描　述
Ctrl+f	屏幕向下移动一页，相当于 Page Down 按钮
Ctrl+b	屏幕向上移动一页，相当于 Page Up 按钮
0	将光标移到该文本的第一个字符处
$	将光标移到该行的第一个字符处
G	将光标移到该文本的最后一行
/word	自光标向下搜索名字为 word 的字符串
?word	自光标向上搜索名字为 word 的字符串
:n1,n2s/word1/word2/g	在 n1 行与 n 2 行之间搜索名字为 word1 的字符串，并将其替换为 word 2
:1,$s/word1/word2/g	在第 1 行与最后一行之间搜索名字为 word1 的字符串，并将其替换为 word 2

点评与拓展：本小节所介绍的 vi 文本编辑器的内容不求全面，但求实用，更多的 vi 用法有待读者在实践中体会。

1.3.2　文件和目录操作

在多数操作系统中都有文件的概念。文件是 Linux 用来存储信息的基本结构，它是被命名(称为文件名)的存储在某种介质(如磁盘、光盘和磁带等)上的一组信息的集合。Linux 文件均为无结构的字符流形式。文件名是文件的标识，它由字母、数字、下划线和圆点组成的字符串来构成。用户应选择有意义的文件名。Linux 要求文件名的长度限制在 255 个字符以内。

为了便于管理和识别，用户可以把扩展名作为文件名的一部分，文件名与扩展名之间用圆点分开，扩展名对于文件分类是十分有用的。用户可能已经对某些大众已接纳的标准

扩展名比较熟悉了，例如，C 语言编写的源代码文件总是具有 C 的扩展名。用户可以根据自己的需要，加入自己的文件扩展名。

在计算机系统中存有大量的文件，如何有效地组织与管理它们，并为用户提供一个使用方便的接口是文件系统的一大任务。Linux 系统以文件目录的方式来组织和管理系统中的所有文件。所谓文件目录就是将所有文件的说明信息采用树型结构组织起来，即我们常说的目录。也就是说，整个文件系统有一个"根"，然后在根上分"杈"，任何一个分杈上都可以再分杈，杈上也可以长出"叶子"。"根"和"杈"在 Linux 中被称为"目录"或"文件夹"，而"叶子"则代表一个个文件。实践证明，这种结构的文件系统处理效率比较高。

对文件进行访问时，需要用到"路径"(Path)的概念。何谓路径？顾名思义，路径是指从树型目录中的某个目录层次到某个文件的一条道路。路径的主要构成是目录名称，中间用"/"符号分开。任一文件在文件系统中的位置都是由相应的路径决定的。

用户在对文件进行访问时，要给出文件所在的路径，这又分相对路径和绝对路径。相对路径是从用户工作目录开始的路径；绝对路径是指从"根"开始的路径，也称为完全路径。

应该注意到，在树型目录结构中到某一确定文件的绝对路径和相对路径均只有一条。绝对路径是确定不变的，而相对路径则随着用户工作目录的变化而不断变化。这一点对于用户以后使用某些命令如 cp 和 tar 等大有好处。

理解了文件、目录和路径的概念后，下面通过简单的例子来讲述文件的复制、移动以及文件目录权限的控制。

cp 命令可以将给出的文件或目录复制到另一文件或目录中，就如同 DOS 下的 copy 命令一样，功能非常强大。输入下面的命令则将 testvi 这个文件复制到/home/tian 目录下，如图 1-35 所示。

```
[root@sec ~]# cp testvi /home/tian/
[root@sec ~]# ls /home/tian
testvi
[root@sec ~]#
```

图 1-35 复制单个文件

```
cp testvi /home/tian
```

这里 testvi 就是相对路径，因为 shell 所在的路径是/root。后面的/home/tian 为绝对路径。

cp 命令的-r 参数提供目录复制功能，- r 参数说明若给出的源文件是一目录文件，则将递归复制该目录下所有的子目录和文件。此时目标文件必须为一个目录名。比如说，要将 nfs 这个目录复制到/home/tian 下，可以使用下面命令，如图 1-36 所示。

```
[root@sec ~]# ls nfs
hostgen  sshrrun  sshrrunon
[root@sec ~]# cp -r nfs /home/tian
[root@sec ~]# ls /home/tian
nfs  testvi
[root@sec ~]# ls /home/tian/nfs
hostgen  sshrrun  sshrrunon
```

图 1-36 复制整个目录

```
cp -r nfs /home/tian
```

从结果可以看到，在/home/tian 下有了 nfs 这个目录，并且 nfs 中的文件也一起复制过去了。如果只需将 nfs 目录下的文件复制到/home/tian 下，而不需要将 nfs 目录也一起复制过去，该如何操作呢？这时就应使用下面的命令：

```
cp nfs/* /home/tian
```

上面命令中的“*”为通配符，意思是将 nfs 目录下的所有文件都复制到/home/tian 下，如图 1-37 所示。

```
[root@sec ~]# cp nfs/* /home/tian
[root@sec ~]# ls /home/tian
hostgen  nfs  sshrrun  sshrrunon  testvi
[root@sec ~]#
```

图 1-37　复制目录下的所有文件

mv 命令可以为文件或目录改名或将文件由一个目录移到另一个目录中。mv 命令中第二个参数类型分目标文件和目标目录，如果类型是文件时，mv 命令将所给的源文件或目录重命名为给定的目标文件名，此时，源文件只能有一个(也可以是源目录)；如果是已存在的目录名称时，源文件或目录参数可以有多个，mv 命令将各参数指定的源文件均移至目标目录中。在跨文件系统移动时，mv 先复制，再将原有文件删除，从而链至该文件的链接也将丢失。

图 1-38 演示了使用 mv 重命名文件和移动文件的方法，第一条命令：

```
mv testvi firstvi
```

将 testvi 这个文件重命名为 firstvi。

第二条命令：

```
mv firstvi /home/tian/
```

将 firstvi 这个文件移动到/home/tian 目录下，相当于 Windows 下的剪切和粘贴操作。

```
[root@sec ~]# mv testvi firstvi
[root@sec ~]# ls
anaconda-ks.cfg  Desktop   firstvi  hangup.sh  install.log
bras.linux.sh    first.sh  gt4      hoh        install.log.syslog
[root@sec ~]# mv firstvi /home/tian/
[root@sec ~]# ls /home/tian
firstvi  hostgen  nfs  sshrrun  sshrrunon
[root@sec ~]#
```

图 1-38　mv 命令的用法

rm 命令提供删除文件功能，该命令可以删除一个目录中的一个或多个文件或目录，它也可以将某个目录及其下的所有文件及子目录均删除。删除单个文件不用带任何参数；如果是删除整个目录及目录下的所有文件，需要带-rf 参数，如图 1-39 所示。这两种命令分

别如下。

(1)　删除 firstvi 文件。

```
rm firstvi
```

(2)　删除 nfs 目录以及该目录下所有文件：

```
rm -rf nfs
```

```
[root@sec tian]# ls
firstvi  hostgen  nfs  sshrrun  sshrrunon
[root@sec tian]# rm firstvi
rm: 是否删除一般空文件"firstvi"? y
[root@sec tian]# ls
hostgen  nfs  sshrrun  sshrrunon
[root@sec tian]# rm -rf nfs
[root@sec tian]# ls
hostgen  sshrrun  sshrrunon
[root@sec tian]#
```

图 1-39　rm 命令用法

Linux 系统中的每个文件和目录都有访问许可权限，用它来确定用户能以何种方式对文件和目录进行访问和操作。

文件或目录的访问权限分为只读、只写和可执行三种。以文件为例，只读权限表示只允许读其内容，而禁止对其做任何的更改操作；可执行权限表示允许将该文件作为一个程序执行。文件被创建时，文件所有者自动拥有对该文件的读、写和可执行权限，以便于对文件的阅读和修改。用户也可根据需要把访问权限设置为任何组合。

有三种不同类型的用户可对文件或目录进行访问：文件所有者、同组用户和其他用户。所有者一般是文件的创建者，他可以允许同组用户访问文件，还可以将文件的访问权限赋予系统中的其他用户，从而使系统中每一位用户都能访问该所有者拥有的文件或目录。

每一文件或目录的访问权限都有三组，每组用三位表示，分别为文件属主的读、写和执行权限，与属主同组的用户的读、写和执行权限，系统中其他用户的读、写和执行权限。当用 ls -l 命令显示文件或目录的详细信息时，最左边的一列为文件的访问权限。如图 1-40 所示，列出了 testvi 这个文件的详细属性，为：

```
-rw-r--r-- 1 root root 0 10-27 13:05 testvi
```

r 代表只读，w 代表写，x 代表可执行。注意这里共有 10 个字符。第一个字符指定了文件类型。在通常意义上，一个目录也是一个文件。如果第一个字符是横线，表示是一个非目录的文件；如果是 d，表示是一个目录。这行就表示 testvi 是一个普通文件，testvi 的属主有读写权限，与 testvi 属主同组的用户只有读权限，其他用户也只有读权限。

```
[root@sec ~]# ls -l testvi
-rw-r--r-- 1 root root 0 10-27 13:05 testvi
[root@sec ~]#
```

图 1-40　查看 testvi 权限

理解了文件的三种权限、三个用户组的概念后，我们来学习如何改变文件的属性，chmod 命令提供了该功能。chmod 命令是 Linux 中非常重要的命令，用来控制文件或目录的访问权限。它有两种用法：一种是包含字母和操作符表达式的文字设定法；另一种是包含数字的数字设定法。文字设定法的格式为：

```
chmod [用户类型] [符号] [属性] 文件名
```

三种参数如表 1-3 所示。

表 1-3　chmod 命令

用户类型	符　号	属　性
u 表示“用户”(user)，即文件或目录的所有者	+ 添加某个权限	r 可读
g 表示“同组”(group)，即与文件属主同组的用户	- 取消某个权限	w 可写
o 表示“其他”(others)用户	= 赋予给定权限并取消其他所有权限	x 可执行
a 表示“所有”(all)用户。它是系统默认值		

例如，如果要将 testvi 这个文件的权限改成：属主可读可写可执行，同组用户可读可写，只需使用下面命令：

```
chmod u+x,g+w testvi
```

注意该命令的不同参数之间用逗号隔开，结果如图 1-41 所示。

```
[root@sec ~]# chmod u+x,g+w testvi
[root@sec ~]# ls -l testvi
-rwxrw-r-- 1 root root 0 10-27 13:05 testvi
[root@sec ~]#
```

图 1-41　chmod 文字设定法

那么什么是 chmod 的数字设定法呢？首先需要了解用数字表示的属性的含义：0 表示没有权限，1 表示可执行权限，2 表示可写权限，4 表示可读权限，然后将其相加。所以数字属性的格式应为 3 个从 0～7 的八进制数，其顺序是 u、g、o。比如上步的 chmod u+x,g+w testvi 命令等价于下面的命令：

```
chmod 764 testvi
```

结果如图 1-42 所示。

点评与拓展： 设置文件权限方面的命令还有 chgrp、chown 等，由于使用较少，在此就省略不讲了。对于 chmod 的两种设定方法，读者只需熟悉其中的一种即可。

```
[root@sec ~]# chmod 764 testvi
[root@sec ~]# ls -l testvi
-rwxrw-r-- 1 root root 0 10-27 13:05 testvi
[root@sec ~]#
```

图 1-42　chmod 数字设定法

1.3.3　shell 脚本

何谓 shell？shell 是“外壳”的意思，很形象，shell 是 Linux 系统内核与用户交互的外壳，它接受用户的命令，然后将命令转达给 Linux 内核执行，内核再将执行结果通过 shell 返回给用户。因此，我们将 Linux 的命令提示行称为 shell 命令行，输入其中的命令称为 shell 命令。那么什么是 shell 脚本呢？简言之，shell 脚本是批量执行 shell 命令的程序，它将许多条 shell 命令整合在一个脚本文件中，执行 shell 脚本就相当于连续执行了整合在其中的许多条 shell 命令。shell 脚本这种程序与 C、Java 等程序不同，它不需要编译，也没有复杂的语法和流程，但却具备良好的 debug 工具。

shell 脚本以简洁的方式提供了批 shell 命令的处理，大大方便了网络管理员对各类 Linux 服务器的配置和架设，因此 shell 脚本成为用户在架设 Linux 服务器前必须了解的概念。下面通过一个例子来简述 shell 脚本的编写和执行的步骤。

❶ 使用 vi 命令编辑文件 first.sh，在该文件内输入如图 1-43 所示的内容。第一行“#!/bin/bash”是每个 shell 脚本必须要有的声明，它宣告该文件是 shell 脚本程序。其他行如果以“#”符号开头，则表明为注释行。

下面三行:

```
echo "Hello World! "
echo "This is My First Shell Script! "
date
```

分别为三条 shell 命令，前两行要求回显引号里的内容，最后一行 date 命令要求显示当前的系统时间。

```
[root@sec ~]# vi first.sh
#!/bin/bash

echo "Hello World! "
echo "This is My First Shell Script! "
date
~
```

图 1-43　第一个 shell 脚本

❷ 存储 first.sh 文件后，还需要使用下面的命令将 first.sh 文件的权限改为可执行，如图 1-44 标注部分所示。这样就可以执行 first.sh 这个脚本文件了，输入./first.sh，结果如图 1-44 所示，屏幕显示了写在 echo 命令后面的语句，并显示出当前的系统时间，所用到的两条命令分列

于下：

```
chmod a+x first.sh
./first.sh
```

```
[root@sec ~]# chmod a+x first.sh
[root@sec ~]# ./first.sh
Hello World!
This is My First Shell Script!
2007年 10月 26日 星期五 21:18:04 CST
[root@sec ~]#
```

图 1-44　执行 first.sh 脚本

❸ shell 脚本可以通过定义变量来实现与用户之间的交互，shell 命令规定以“$”符号开头的字符串为变量名，变量可以用来赋值传递；与 C 语言变量不同的是，shell 的变量无需定义类型。我们写一个带变量的 shell 脚本，它接受用户的输入，然后将输入反馈输出，脚本内容如图 1-45 所示。read 命令读取来自键盘的输入，将输入赋给变量；echo 命令反馈字符串时，以$firstname 和$lastname 表示读取刚才赋过值的两个变量，执行效果如图 1-46 所示。

```
[root@sec ~]# vi first.sh
#!/bin/bash

read -p "Please input your first name " firstname
read -p "Please input your last name " lastname
echo "Your full name is: $firstname $lastname"
```

图 1-45　带变量的 shell 脚本

```
[root@sec ~]# ./first.sh
Please input your first name Jonghun
Please input your last name Park
Your full name is: Jonghun Park
[root@sec ~]#
```

图 1-46　执行效果

点评与拓展：shell 脚本的编写是一个比较复杂的问题，不是预备知识中的少量篇幅所能介绍清楚的，这里只是介绍 shell 脚本的基本概念和简单脚本的编写方法，为读者理解后面章节中配置各类服务器时所出现的脚本打下基础。

1.3.4　系统用户管理

安装 Fedora 时，设置过 root 用户的密码，root 用户是 Linux 的超级用户，可以创建和删除其他所有系统用户。本小节将讲述如何创建和删除系统用户、如何修改系统用户的密码以及如何在用户之间切换等内容。

一般都使用 Linux 提供的命令 useradd 来添加新用户，输入下面的命令添加 Foxdie 用户，如图 1-47 所示。

```
[root@sec ~]# useradd Foxdie
[root@sec ~]# ls -l /home
总计 848
drwxr-xr-x  3 Foxdie     Foxdie        4096 10-27 14:20 Foxdie
drwx------  3 ftptest    ftptest       4096 10-11 20:50 ftptest
drwxr-xr-x  4 globus     globus        4096 2007-04-10 globus
```

图 1-47　添加 Foxdie 用户

```
useradd Foxdie
```

创建新用户看似很简单，其实已经在系统里创建了很多东西，该命令默认地在/home 下为 Foxdie 用户创建了根目录，如图 1-47 标注部分所示。如果要将 Foxdie 的根目录创建在别的目录下，可以在 useradd 后面带-d 参数来指定。

系统下其他关于用户的文件还有/etc/passwd、/etc/shadow、/etc/group，在使用 useradd 命令创建 Foxdie 后，这些文件也相应地创建了 Foxdie 的相关信息，如图 1-48 所示。

```
[root@sec ~]# grep Foxdie /etc/passwd /etc/shadow /etc/group
/etc/passwd:Foxdie:x:506:506::/home/Foxdie:/bin/bash
/etc/shadow:Foxdie:!!:13813:0:99999:7:::
/etc/group:Foxdie:x:506:
[root@sec ~]#
```

图 1-48　相关文件中的 Foxdie 信息

出于系统安全考虑，Linux 系统中的每一个用户除了有其用户名外，还有其对应的用户口令。因此使用 useradd 命令创建新用户后，还需使用 passwd 命令为每一位新增加的用户设置口令。root 用户可以使用 passwd 命令来改变系统用户的口令，系统用户也可以使用 passwd 命令来改变自己的口令。图 1-49 演示了 root 用户为 Foxdie 用户更改密码的过程，命令如下：

```
passwd Foxdie
```

```
[root@sec ~]# passwd Foxdie
Changing password for user Foxdie.
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
[root@sec ~]#
```

图 1-49　为 Foxdie 用户更改密码

口令被加密并放入/etc/shdo 文件。选取一个不易被破解的口令是很重要的。应遵守如下规则：口令应该至少有 6 位(最好是 8 位)字符；口令应该是大小写字母、标点符号和数字混杂的。

su 命令提供了用户之间的切换功能，这个命令非常重要，它可以让一个普通用户拥有超级用户或其他用户的权限，也可以让超级用户以普通用户的身份来操作。超级用户切换

到普通用户时，无需输入密码；反之，则需要输入超级用户的密码。使用下面的命令，将由 root 切换到 Foxdie 用户，如图 1-50 所示。

```
su Foxdie
```

在 Foxdie 下要更改自己的密码，可直接输入 passwd 命令，此时需要输入旧密码，然后才能输入新密码，如图 1-50 所示。

```
[root@sec ~]# su Foxdie
[Foxdie@sec root]$ passwd
Changing password for user Foxdie.
Changing password for Foxdie
(current) UNIX password:
New UNIX password:
```

图 1-50　切换到 Foxdie 用户

使用命令 userdel 可以删除系统用户，如果带-r 参数，表示连系统用户的根目录一起删除。具体命令如下：

```
userdel -r Foxdie
```

图 1-51 标注部分表示删除 Foexdie 用户，并且连同 Foxdie 的根目录一起删除。

```
[root@sec ~]# userdel -r Foxdie
[root@sec ~]#
```

图 1-51　删除 Foxdie 用户

1.4　本 章 小 结

本章首先介绍了 Linux 的起源和优点、Fedora 与 Linux 的关系及 Fedora 版本的发布等内容；接着重点介绍了 Fedora 6 的安装过程，详细讲述了安装过程中的磁盘分区步骤；最后介绍了后续章节需要用到的 Linux 系统的一些基本知识：例如，vi 文本编辑器，它是创建文本文件、编辑文本文件所必须使用的工具；文件和目录的复制、移动、删除，以及文件和目录权限的管理也贯穿于全书；shell 脚本，它提供了处理批 shell 命令的一种方法，方便了网络管理员配置各种服务器；Linux 系统用户的创建、密码的更改及用户的删除，这些也是架设服务器中不可避免的问题。本章在介绍这些预备知识时，都不求全面，只求能够为后续章节的学习打下基础。

第 2 章　Linux 服务器基本网络配置

Internet 利用 TCP/IP 协议族进行网际互联，TCP/IP 协议族是由分布在网际互联各个层次的一系列协议所组成的。在学习 Linux 各种服务器的架设前，需要对 TCP/IP 协议族中的主要协议及一些重要概念有所了解。本章首先将回顾 TCP/IP 协议族，重点介绍 IP 地址和 MAC 地址、子网划分和子网掩码、网络层路由及 TCP、UDP 协议等内容；然后讲述 Linux 连接 Internet 的基本网络配置，包括局域网内的各个网络参数的设定和如何利用 ADSL 连接 Internet；最后讲述如何在单网卡的服务器上架设路由。

通过本章的学习，读者应掌握以下内容：

✧ 了解 TCP/IP 体系架构和各层中协议的基本功能
✧ 深入理解子网划分和子网掩码以及网关路由的概念
✧ 能够对 Linux 进行基本的网络配置
✧ 了解利用 ADSL 连接 Internet 的配置方法
✧ 能够架设 Linux 服务器路由

2.1　TCP/IP 协议族概述

2.1.1　TCP/IP 体系架构

TCP/IP 是一组网络协议，TCP 和 IP 是这一组协议中的两个最重要的协议。协议就好像是一种语言，有一定的规则，TCP/IP 是一组计算机使用的规则，它使得运行不同操作系统的计算机能以有序的方式交换数据。最常用于描述数据通信的模型是开放系统互联模型(OSI 模型)，它给出了理论上的七层模型：应用层、表示层、会话层、传输层、网络层、数据链路层和物理层。美国国防部从实践的角度精简了这七层模型，提出了 TCP/IP 四层体系结构，包含应用层、传输层、网络层和网络接口层，如图 2-1 所示。TCP/IP 协议族中的每一个协议规范均定义在 RFC 中，RFC 由许多 Internet 用户提交，这些用户可以提出新的协议，可以建议对现有协议进行改进，甚至对网络状态进行评价。

✧ 应用层是体系结构中的最高层，它确定了进程间通信的性质以满足用户需要。应用层中的协议很多，如 HTTP 协议、SMTP 协议、FTP 协议等。
✧ 传输层负责主机中两个进程间的通信，它有两种不同协议，一是面向连接的 TCP(transmission control protocol，传输控制协议)，一是无连接的 UDP(user datagram protocol，用户数据包协议)。

应用层 (FTP、Telnet、SMTP、NFS、DNS等协议)
传输层(TCP、UDP协议)
网络层(IP、路由协议、ICMP)
网络接口层(ARP、RARP)

图 2-1　TCP/IP 体系架构

✧ 网络层负责为分组交换网上的不同主机提供通信能力。网络层另外还有路由选择功能，使源主机的数据包能传到目的主机。网络层协议主要是无连接的 IP(Internet Protocol，网际协议)和多种路由选择协议。网络层又称网际层或 IP 层。
✧ 网络接口层是 TCP/IP 的最底层，负责数据包的接收或发送。另外，也有人将网络接口层再划分为两层，即网络接口层和物理层，从而构成五层体系结构。

TCP/IP 的优点如下：①一开始就考虑到多种异构网的互联问题，并将网际协议 IP 作为 TCP/IP 的重要组成部分；②面向连接服务和无连接服务并重；③较好的网络管理功能。TCP/IP 也有不足之处，TCP/IP 模型没有清晰区分“服务”、“协议”、“接口”等概念，可能为采用新技术设计新网络带来一些麻烦；TCP/IP 模型的通用性较差，难以用来描述其他类协议栈；TCP/IP 的网络接口层严格地说不是一个层次而仅是一个接口。

2.1.2　网际 IP 协议

本节首先介绍 TCP/IP 体系架构中网络层的地址，即所谓的 IP 地址，以及网络接口层的网卡地址，即 MAC 地址。在 IP 地址的基础上介绍网段的划分以及子网掩码等重要概念。

1. IP 地址和 MAC 地址

首先，需要区分 IP 地址和 MAC 地址(硬件地址)的概念。IP 地址是网络层及其以上各层使用的地址，MAC 地址是数据链路层和物理层使用的地址。IP 地址放在 IP 数据包的首部，MAC 地址放在 MAC 帧的首部。当 IP 数据包放入数据链路层的 MAC 帧中以后，IP 数据包就成了 MAC 帧的数据，在数据链路层看不见数据包的 IP 地址。

IP 地址和 MAC 地址的主要区别如下：

✧ IP 层抽象的 Internet 上只能看到 IP 数据包。IP 数据包不管经过多少路由器的转发，其首部中的源地址和目的地址始终不变。
✧ 物理网络的链路层只能看到 MAC 帧。MAC 帧在不同网络上传送时，其 MAC 帧首部中的源地址和目的地址要发生变化。但 MAC 帧首部的变化 IP 层是看不见的。

虽然互联在一起的 MAC 地址体系各不相同，但 IP 层抽象的 Internet 屏蔽了下层的复杂细节，只要在网络层上讨论问题，就能使用统一、抽象的 IP 地址。

2．IP 数据包

IP 数据包是一种数据报文的格式。IP 数据包最大可以达到 65 535 字节(Byte)，而标准以太网帧包含的数据最大仅有 1 500 字节，并且不同网络媒体的 MAC 帧大小也不同。由于 IP 数据包必须放到 MAC 帧中，因此 Internet 上的 IP 数据包必须小于 MAC 帧所能允许的最大值，而不可能达到它的最大值(65 535 字节)。此外，在网络联机过程中，数据包所经过的网络媒体各不相同，MAC 帧的大小也就随之变化。为使 IP 数据包适用于所有的网络，IP 数据包是可以重组的。

MAC 帧表头中最重要的就是网卡号，而 IP 数据包表头中最重要的就是 IP 地址。目前 Internet 上使用的 IPv4 的 IP 地址是由 32 位二进制数据所组成。由于人们对二进制数不很熟悉，因此就将 32 位的 IP 地址分成 4 小段，每一小段含 8 位，再将每小段的 8 位换算成十进制，并且每一小段中间以小数点隔开，就成了目前大家所熟悉的 IP 地址书写格式。

3．网段与 IP 地址分类

IP 地址的 32 位数据，主要分为主机号(HOST_ID)和网络号(Net_ID)两部分。下面以 192.168.10.0～192.168.10.255 这一 C 类网段为例进行说明，如图 2-2 所示。

```
192.168.10.0~192.168.10.255  C 类网段：
11000000.10101000.00001010.00000000
11000000.10101000.00001010.11111111
|----------网络号---------|-主机号-|
```

图 2-2　C 类网段的例子

在本例中，前面三组数字(即 192.168.10)称为网络号，最后一组数字称为主机号。“同一个网段中”的定义是指：在同一个物理网段内，主机的 IP 地址具有相同的网络号，并且具有不同的主机号。如果满足上述条件，这些 IP 地址就是同一个网段内的 IP 地址。

上面例子中的 192.168.10.1、192.168.10.2、…、192.168.10.255 这些 IP 地址就是同一个网段内的 IP 地址。需要注意的是，同一个网段内(网络号相同)，不能具有相同的主机号，否则就会发生 IP 地址冲突，造成两台主机都无法连接网络。那么设定同一个网段时要注意哪些问题呢？将 IP 地址设定在同一个网段内又有什么好处呢？

- 在同一个网段内，网络号是不变的，而主机号则是不可重复的。另外，主机号在二进制表示法中，不可同时为 0，也不可同时为 1。在上面的例子中，192.168.10.0(主机号全部为 0)和 192.168.10.255(主机号全部为 1)就不可以用作该网段内主机的 IP 地址，即这个网段内可以用来设定主机的 IP 地址范围是 192.168.10.1～192.168.10.254。
- 在同一个网络内，每一台主机都可以以 MAC 帧的格式传递数据。在通过 ARP 协议和广播数据包取得 MAC 地址与 IP 地址的对应后，直接利用 MAC 帧传递数据。
- 在同一个物理网段内，如果两台主机设定成不同的 IP 网段，则两台主机无法直接以 MAC 帧格式进行数据传递，因为广播数据包无法查询到 MAC 地址与 IP 地址的对应。

✧ 当主机号所占用的数据位越多，也即主机号数量越多时，就表示同一个网段内可用以设定主机的 IP 地址数量越多。

因此，一般企业内的所有计算机都设定在同一个网段内是最方便的。这样，每一台计算机都可以直接通过 MAC 帧来进行数据传送，而不必经过路由器来进行数据包的转递。

网络号越大，表示主机号越少，也即网段内可以分配的 IP 地址数量就越少。这说明网络号是有分级的。目前 Internet 将 IP 地址简单地分成三种常见的类型，即所谓 A、B、C 三类地址，它们各自的意义如图 2-3 所示。

```
以二进制说明 Network 第一个数字的定义：
A 类 : 0xxxxxxx.xxxxxxxx.xxxxxxxx.xxxxxxxx  网络号开头是 0
      |-网络号-|---------主机号----------|
B 类 : 10xxxxxx.xxxxxxxx.xxxxxxxx.xxxxxxxx  网络号开头是 10
      |-----网络号------|-----主机号------|
C 类 : 110xxxxx.xxxxxxxx.xxxxxxxx.xxxxxxxx  网络号开头是 110
      |---------网络号----------|-主机号-|
```

图 2-3　A、B、C 三类 IP 地址

三种分级的十进制表示如下：

A 类 0.xx.xx.xx～126.xx.xx.xx

B 类 128.xx.xx.xx～191.xx.xx.xx

C 类 192.xx.xx.xx～223.xx.xx.xx

细心的读者可能会发现 127.xx.xx.xx 没有列出，它照理也应是 A 类地址的一段。的确如此，事实上这个网段被操作系统用作内部循环网络测试了。在各个操作系统中，不管该主机的硬件有没有网卡，为了让作业确认自己的网络没有问题，便将 127.xx.xx.xx 这个 A 类地址的网段用到操作系统上，作为内部的回路测试。因此，像 127.0.0.1 就不可以再用作其他网卡的网段设定。

4．子网掩码与子网划分

前面提到的 A、B、C 三类 IP 地址是由 IP 协议预设分配的。在这几类地址中，不难发现：A 类地址可用于设定计算机主机 IP 地址的数量最多(主机号最大)。在同一个 A 类网段内，最大主机数量可以达到 $2^{24}-2$ 个。但是，达到最大主机数量是会影响网络性能的。

在共享的网络链路上，任何一台主机想要使用时，就得使用CSMA/CD(载波监听多点接入/冲突检测)的方式去进行网络监听工作。对于这样大的一个网络架构来说，每台主机发出 MAC 帧前要进行的CSMA/CD会造成系统很严重的停顿问题。因为在冲突检测和进行 MAC 地址与 IP 地址对应的广播时，要响应的主机数量太多，这样整个网络的性能就会变得很差。因此，一般设定 C 类地址作为整个局域网的架构，事实上连 C 类地址也都显得太大了。一个网络内的主机要不超过 30 台，网络的性能才会比较好。

当然还可以对 C 类网络继续进行划分。32 位 IP 地址分为网络号与主机号两部分，C 类地址的网络号占了 24 位，而且还可以划分得更细。让第一个主机号位用作网络号，整个网络号就有了 25 位，主机号则减少为 7 位。这样，原来的一个 C 类网络就被划分为两个子网，每个子网有 126(256/2−2)个可用的 IP 地址。如此一来，该网络中的主机在进行逻辑广播时，响应的主机数量就减少了一半，这对于网络的性能是有好处的。

那么，到底是什么参数实现子网划分的呢？这就是子网掩码(Subnet Mask)。子网掩码是用来定义网络的最重要的一个参数。而子网掩码的设定依据又是什么呢？继续以192.168.10.0～192.168.10.255 这个网段为例，该类 IP 地址可以分为网络号和主机号两部分，由于网络号是不可变的，则假设它所占据的数据位都已被用光(全部为 1)，而主机号是可变的，则将它假设成是保留的(全部为 0)，子网掩码的表示如图 2-4 所示。

```
192.168.10.0~192.168.10.255  的子网掩码表示
11000000.10101000.00001010.00000000
11000000.10101000.00001010.11111111
|----------网络号---------|-主机号-|
11111111.11111111.11111111.00000000  子网掩码（二进制）
  255   .  255   .  255   .   0      子网掩码（十进制）
```

图 2-4 子网掩码示例

同理，A、B、C 三类地址的子网掩码的表示如图 2-5 所示。

```
A, B, C 三类地址的子网掩码表示方式：
       |---------------二进制---------------|------十进制------|
A 类 : 11111111.00000000.00000000.00000000   255.  0.  0.  0
B 类 : 11111111.11111111.00000000.00000000   255.255.  0.  0
C 类 : 11111111.11111111.11111111.00000000   255.255.255.  0
```

图 2-5 A、B、C 三类地址的子网掩码

因此，192.168.10.0～192.168.10.255 这个 C 类网段的子网掩码就是 255.255.255.0。当主机号全部为 0 或全部为 1 时，该 IP 地址是不可以使用的。因为主机号全部为 0 时，表示这个 IP 地址是该网段的网络地址；全部为 1 时，则表示该网段的最后一个 IP 地址被用作广播地址。因此，192.168.10.0～192.168.10.255 这个网段里的相关网络参数就有如下几个。

✧ 子网掩码：255.255.255.0　　网段定义中最重要的参数
✧ 网络地址：192.168.10.0　　第一个 IP 地址
✧ 广播地址：192.168.10.255　　最后一个 IP 地址
✧ 可用来设定为主机的 IP 地址数：192.168.10.1～192.168.10.254

一般来说，如果知道了网络地址和子网掩码，就可以定义出该网络的所有 IP 地址。因为由子网掩码就可以推出广播的 IP 地址。所以，常常会以网络地址和子网掩码来表示一个网络，例如，下面的写法。

网络地址/子网掩码：192.168.10.0/255.255.255.0

192.168.10.0/24 (因为网络号共有 24 位)

由于子网掩码里的网络号都是 1，而 C 类地址共有 24 位网络号，所以就有192.168.10.0/24 的写法，这也是一般网络的表示方法。前面提到 C 类网络还可以继续进行子网划分，就以 192.168.10.0/24 这个网络为例，看看如何再细分为两个子网。既然主机号可以用来当作网络号，那么当网络号使用了 25 位时，就可以划分为两个子网了，如图 2-6 所示，而且两个子网还可以这样再细分下去。

```
原来的 C 类地址网络号与主机号的划分：
11000000.10101000.00001010.00000000    网络地址：  192.168.10.0
11000000.10101000.00001010.11111111    广播地址：192.168.10.255
|----------网络号---------|-主机号-|

划分成两个子网之后的网络号与主机号：
11000000.10101000.00001010.0 0000000  多了一个网络号，为 0
11000000.10101000.00001010.1 0000000  多了一个网络号，为 1
|----------网络号-----------|-主机号-|

第一个子网
网络地址：11000000.10101000.00001010.0 0000000   192.168.10.0
广播地址：11000000.10101000.00001010.0 1111111   192.168.10.127
         |-----------网络号-----------|-主机号-|
子网掩码：11111111.11111111.11111111.1 0000000   255.255.255.128
网段表示方式：
192.168.10.0/25 或 192.168.10.0/255.255.255.128

第二个子网
网络地址：11000000.10101000.00001010.1 0000000   192.168.10.128
广播地址：11000000.10101000.00001010.1 1111111   192.168.10.255
         |----------网络号------------|-主机号-|
子网掩码：11111111.11111111.11111111.1 0000000   255.255.255.128
网段表示方式：
192.168.10.128/25 或 192.168.10.128/255.255.255.128
```

图 2-6　子网划分示例

2.1.3　网络层路由简介

以太网在物理网段内可以使用 MAC 帧直接在两张网卡之间传递数据。在 Internet 环境中，同一个网段(网络号相同的网段)内可以直接采用广播的方式以 ARP 协议来取得 IP 地址与 MAC 地址的对应，从而使数据可以在同一个网段内进行传输。但是，如果要向不在同一个网段的主机请求数据，这时候数据包该如何传递呢？这就引出了路由(Route)的概念，它是网络层里的一个重要概念。

1．Internet 的路由

所谓非同一个网段就是网络地址/子网掩码不在同一个地址，也就是两个主机间的网络号不同。两个非同一个网段的主机需要互相通信时，就要利用 IP 的路由选择功能。如图 2-7 所示，图中共有两个不同的网段，分别是 LAN1 与 LAN2，这两个网段通过一个路由器(R1)来进行数据转递。当客户机 BOB 想要传送数据到客户机 DARK 时，它的 IP 数据包该如何传输呢？

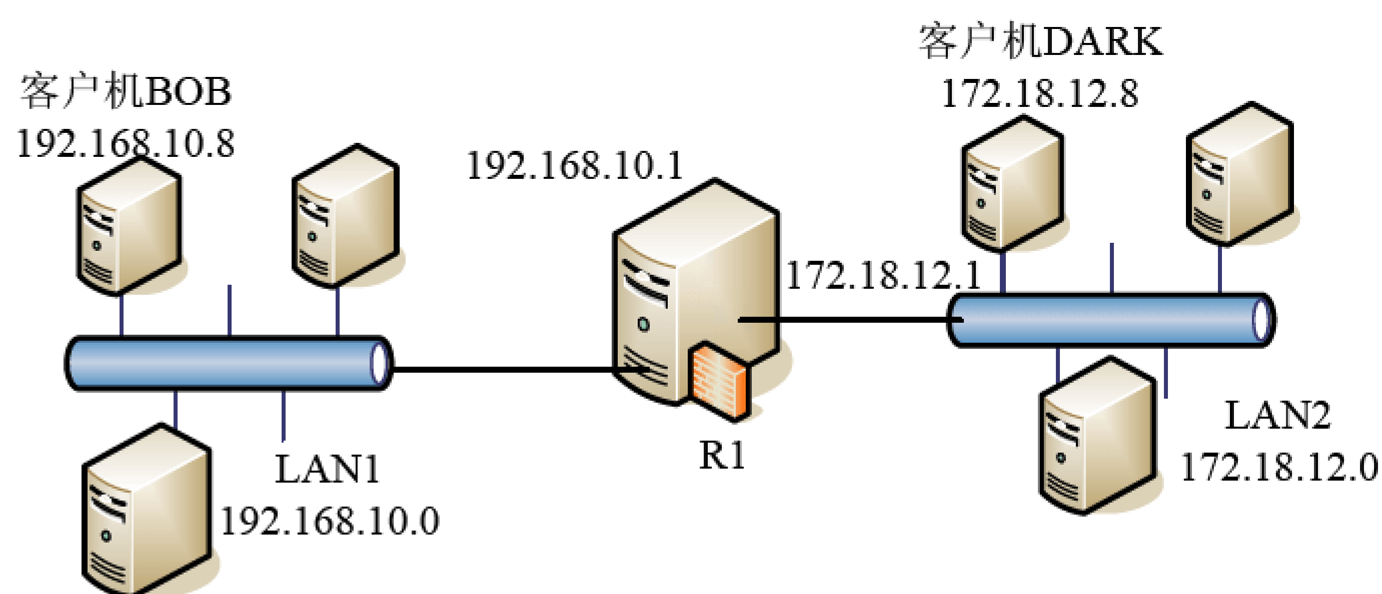


图 2-7　路由器示意图

由于 LAN1(192.168.10.0/24)和 LAN2(172.18.12.0/24)是不同网段，所以 BOB 与 DARK 不能直接互通数据。当然，BOB 和 DARK 是通过网络号来发现它们不在同一个网段内的。当主机想要传送数据时，主要的参考就是路由表，每个主机都有自己的路由表。在本例中，BOB 将数据传送到 DARK 的过程大致如下。

(1)　当 BOB 有 IP 数据包需要传送时，主机首先会查看 IP 数据包表头的目的 IP 地址。

(2)　BOB 接着会分析自己的路由表，当发现目的 IP 地址与本机 IP 地址的网络号相同时(即在同一网段)，则 BOB 将会参考自己的 ARP 记录，直接利用 MAC 帧来传递数据。在本例中，BOB 与 DARK 并不在同一网段，BOB 会分析路由表中是否有相符合的路由设定。如果没有，就直接将该 IP 数据包送到默认网关。本例中的默认网关是 R1。

(3)　当 IP 数据包被送至 R1 后，R1 同样先分析该 IP 数据包的目的地址，然后检查 R1 自己的路由设定。通常，作为路由器的主机 R1，会有两个以上的接口来连接不同的网段。在本例中，R1 会发现 IP 数据包的目的地址是 172.18.12.8，正好是 LAN2 网段的区域。因此 R1 会直接以 MAC 帧将数据传送给 DARK。

每一个主机都会有一个路由表，数据的传递将依据这个路由表进行。一旦数据包依据路由表的规则传送出去后，主机自己就不再管数据包的流向了。该数据包的流向将由下一跳路由器来决定，而下一跳路由器在传递时，也是依据自己的路由表来判断数据包该经由哪里传送出去。

如图 2-8 所示，客户机 Alex 要将数据传送到服务器 Zoom，则依据自己的路由表，先将数据包送到路由器 Fahringer，Fahriger 再继续传送到下一跳路由器，这样一个一个接力传递下去，最后总是可以到达 Zoom 的。

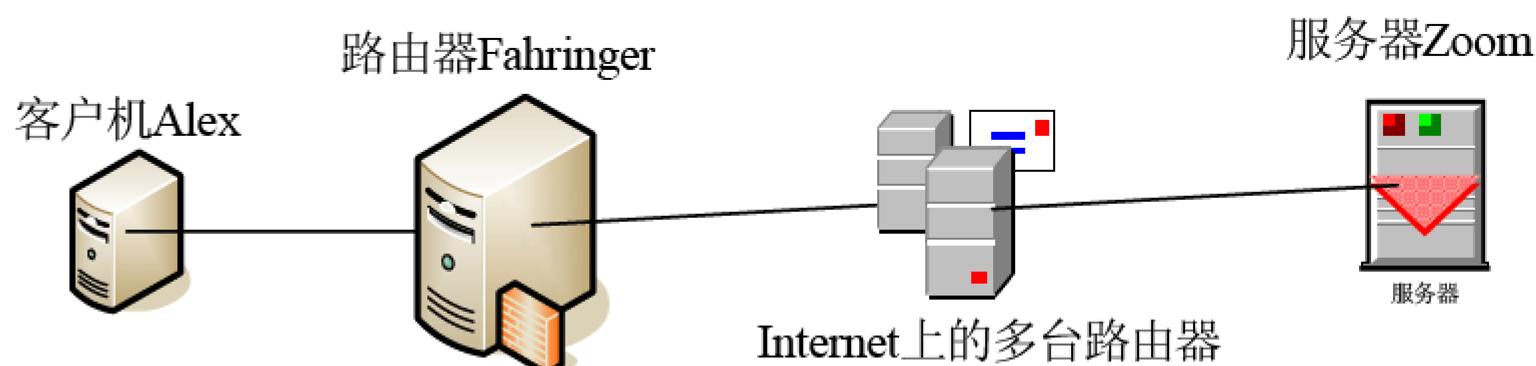


图 2-8　Internet 路由器的传接

上例只是介绍了一种很简单的路由，事实上，Internet 上的路由协议与变化是相当复杂的，因为 Internet 上的路由并不是静态的，它可以随时根据环境变化而修改每个数据包的传送方向。要深入地了解路由可参考其他相关资料。

2．主机路由表的操作

既然路由这么重要，而且一旦设定错误，将会造成某些数据包完全无法正确送出，所以，需要仔细查看主机的路由表(每个主机都有自己的路由表)。查看路由表的指令很简单，就是 route，具体命令格式如图 2-9 所示。

```
[root@cgsp network-scripts]# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
10.1.0.0        *               255.255.255.0   U     0      0        0 eth0
192.168.10.0    *               255.255.255.0   U     0      0        0 ib1
211.65.63.0     *               255.255.255.0   U     0      0        0 eth1
169.254.0.0     *               255.255.0.0     U     0      0        0 eth0
default         211.65.63.1     0.0.0.0         UG    0      0        0 eth1
[root@cgsp network-scripts]#
```

图 2-9　查看路由表

输入 route 指令后，图中的路由表显示这台主机上共有 5 个路由规则。路由表中第一列为目的网络地址，例如，10.1.0.0 就是一个目的网络地址；最后一列是到达目的网络所要使用的网络接口，例如，eth0 就是网卡的装置代号。如果要传送的数据包在 10.1.0.0/255.255.255.0 的路由规则里，由于其网关地址列为“*”，所以会直接通过对应的网络接口传送出去，而不经过网关；如果要传送的数据包目的 IP 地址不在路由规则里，则将数据包按默认的路由规则(即路由表中 default 字段后面的网关地址)传送出去，即通过 211.65.63.1 网关传递。几乎每一台主机都会有一个默认网关来负责所有非同一网段内的数据包转递。

可以添加和删除路由表中的路由。比如：添加新网段 192.168.1.0，则所有到此网段的 IP 包从 eth1 号网卡传输。可以使用下面的命令：

```
route add -net 192.168.1.0 netmask 255.255.255.0 dev eth1
```

结果如图 2-10 所示，路由表第一行出现了刚才所添加的路由。删除命令如下：

```
route del -net 192.168.1.0 netmask 255.255.255.0 dev eth1
```

```
[root@cgsp network-scripts]# route add -net 192.168.1.0 netmask 255.255.255.0 dev eth1
[root@cgsp network-scripts]# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0     *               255.255.255.0   U     0      0        0 eth1
10.1.0.0        *               255.255.255.0   U     0      0        0 eth0
192.168.10.0    *               255.255.255.0   U     0      0        0 ib1
211.65.63.0     *               255.255.255.0   U     0      0        0 eth1
169.254.0.0     *               255.255.0.0     U     0      0        0 eth0
default         211.65.63.1     0.0.0.0         UG    0      0        0 eth1
[root@cgsp network-scripts]#
```

图 2-10　添加路由

执行结果如图 2-11 所示。

```
[root@cgsp network-scripts]# route del -net 192.168.1.0 netmask 255.255.255.0 dev eth1
[root@cgsp network-scripts]# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
10.1.0.0        *               255.255.255.0   U     0      0        0 eth0
192.168.10.0    *               255.255.255.0   U     0      0        0 ib1
211.65.63.0     *               255.255.255.0   U     0      0        0 eth1
169.254.0.0     *               255.255.0.0     U     0      0        0 eth0
default         211.65.63.1     0.0.0.0         UG    0      0        0 eth1
[root@cgsp network-scripts]#
```

图 2-11　删除路由

2.1.4　TCP/IP 常见网络协议简介

1．ARP 和 RARP 协议

IP 地址只是主机在抽象网络层中的地址，不能直接用来通信。若要将网络层中的数据包传送给目的主机，还需要传到链路层转变成 MAC 帧后才能发送到物理网络上。所以不管网络层用的是什么协议，在实际网络链路上传递数据帧时，最终还是必须用硬件地址。IP 地址有 32 位，硬件地址是 48 位，所以它们之间不存在简单的映射关系。一个网络上可能经常会有新的主机加入，或撤走一些主机，更换网卡也会使主机硬件地址改变。因此，主机中应该存放一个从 IP 地址到硬件地址的映射表，并且这个映射表必须能经常地动态更新。ARP(地址解析协议)很好地解决了这个问题。只要主机或路由器要和本网络上另一已知 IP 地址的主机或路由器进行通信，ARP 协议就会自动将该 IP 地址解析为链路层所需要的硬件地址。在地址转换时，有时还要用到逆地址解析协议 RARP。RARP 使只知道自己硬件地址的主机能够知道其 IP 地址，这种主机往往是无盘工作站。

简而言之，ARP 协议实现由 IP 地址查找 MAC 地址，RARP 协议实现由 MAC 地址查找 IP 地址。

2．UDP 协议

UDP 在 IP 的数据包服务的基础上增加了端口功能和差错检测功能。与 TCP 不同，UDP 用户数据包只提供不可靠的交付。UDP 用户数据包的格式如图 2-12 所示。

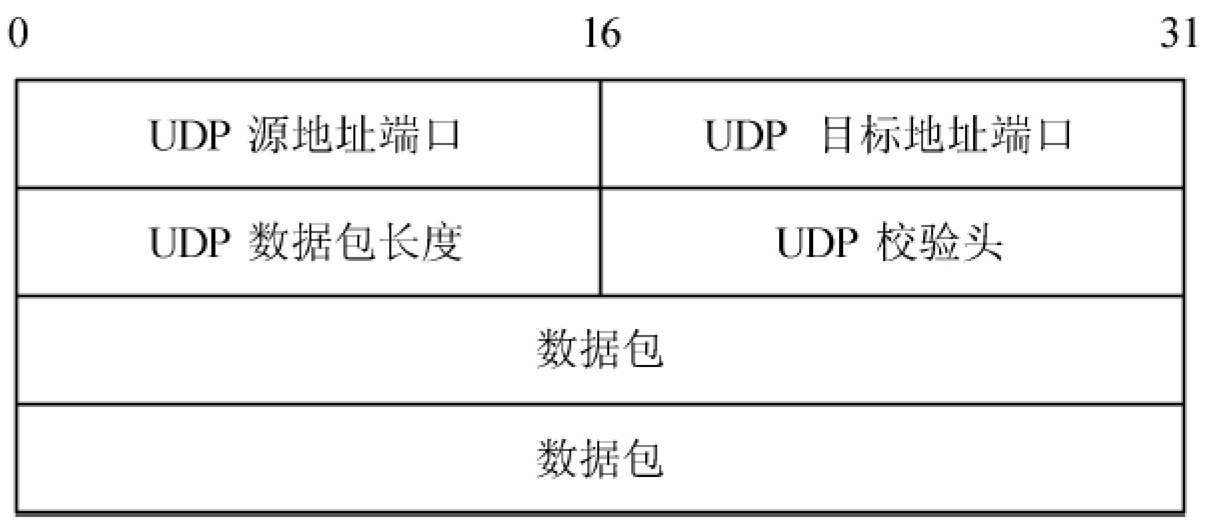


图 2-12　UDP 报文格式

由于不需要确认对方是否已正确接收到数据，所以 UDP 用户数据包的首部比较简单，只有 8 个字节，分 4 个字段，每个字段 2 个字节。UDP 比较适合需要实时反应而数据正确性不是很重要的应用场合，例如，IP 电话、实时视频等。UDP 的优点如下。

- 发送数据前不需要建立连接，因而发送数据结束时也没有连接需要释放，从而减少了开销和发送数据之前的时延。
- UDP 不使用拥塞控制，也不保证可靠支付，因此主机不需要维持具有许多参数的、复杂的连接状态表。
- UDP 用户数据包只有 8 个字节的首部开销，比 TCP 的 20 个字节的首部要短。
- 由于 UDP 没有拥塞控制，因此网络出现的拥塞不会使源主机的发送速率降低。这

对某些实时应用很重要。

3．TCP 协议

TCP 是 TCP/IP 体系中面向连接的传输层协议，它提供全双工和可靠交付的服务。TCP 和 UDP 最大的区别就是：TCP 是面向连接的，UDP 是无连接的。

IP 协议仅负责将数据包传送到正确的目的主机而已，至于目的主机是否真的接收了该数据包就不一定了。要确认该数据是否被目的主机正确接收，必须在数据包上增加一些参数来判断，TCP 报文段(segment)的首部就提供了这样一些参数信息。

如图 2-13 所示就是一个 TCP 报文段的格式，分首部和数据两部分。首部的前 20 个字节是固定的，后面的 4N(N 为整数)字节是长度可变的选项。TCP 的全部功能都体现在其首部中各字段的作用。要掌握 TCP 的工作原理必须先搞清其首部各字段的作用，其中，源端口、目的端口及控制比特是比较重要的字段。

0	4	10	16	24	31
TCP 源地址端口			TCP 目标地址端口		
序 号					
确认号					
数据偏移	保 留	控制比特	窗 口		
校验和			紧急指针		
选 项				填 充	
数 据					
数 据					

图 2-13　TCP 报文格式

了解 TCP 的特点和报文格式后，还需要了解与之密切相关的一个概念：网络端口。端口是操作系统提供的与其他主机进行数据交换的通道，端口的通信是双向的，两台主机通过开启同一个端口使启动的程序实现数据交换。例如，当用户从 FTP 服务器下载资料时，启动 FTP 软件，它通过 21 号端口与 FTP 发送命令进行连接，连接成功后，FTP 软件启动 FTP-DATA，通过 20 号端口传输数据，将 FTP 服务器上的资料下载到本地。

每台主机有 2^{16}(即 65536)个网络端口可供使用，那么应用程序如何选择端口呢？又如何避免端口选择时的冲突呢？为此，Internet 规定了很多流行应用程序的端口号，这些端口号通常小于 1024。当启动新的应用程序，其端口没有设定时，将在大于 1024 的空闲端口中选取。

在 Linux 系统中，/etc/services 配置文件规定了网络应用程序(或者称为网络服务)与端口的对应关系，常用的网络应用程序的固定端口如表 2-1 中所列，在后面章节针对各类服务器的配置中将会用到。

表 2-1　TCP 常用端口与描述

端　口	关 键 字	描　述
20	FTP-DATA	FTP 协议主动数据传输端口
21	FTP	FTP 协议的命令通道
22	SSH	安全的远程连接端口
23	TELNET	早期的远程连接端口
25	SMTP	简单邮件传输协议所使用的端口
53	DNS	域名解析端口
80	WWW	Web 服务器端口
110	POP3	邮件收信协议端口

4．ICMP 协议

为提高 IP 数据包交付成功的机会，网际层使用了因特网控制报文协议 ICMP(Internet Control Message Protocol)。ICMP 允许主机或路由器报告差错情况和提供有关异常情况的报告。ICMP 是因特网标准协议，但不是高层协议，而是网络层协议。ICMP 报文作为 IP 层数据包的数据，加上数据包的首部，组成数据包发送出去。

ICMP 主要完成以下 4 个功能。

✧ 流量控制：当数据接收方太忙时，就不能接收发送方传入的数据流，接收方发送源站抑止报文，以中止数据流的发送。

✧ 不能到达目的地警告：如果网络上的主机检测到其目的是不可达的，不管是因为目的地址与网络上的机器操作不匹配，还是因为连接错误，它都将向发送方发送一个“目的地不可达”的报文。

✧ 重定向路由：网关发送 ICMP 重定向报文，以告诉发送方使用另一个网关。比如，网关 A 接收来自某主机的一个报文，该报文要发送到某一指定的网络，如果网关 A 知道主机如果使用网关 B 能够更快地到达目的网络，网关 A 就将发送一个 ICMP 重定向报文，告诉发送主机将报文发送到网关 B。

✧ 检查远程主机：ICMP 回送报文用于检查互联网络上各主机的连通性，通常又被称为 ping 报文分组。

🕮**点评与拓展：**除了上述介绍的协议，TCP/IP 协议族还包括 IGP、BGP、SMTP 等多种协议，本小节仅介绍与后续章节有联系的几种协议。读者如果需要更深入地了解 TCP/IP 协议族，可以参考其他一些有关的资料。

2.2 Linux 系统的网络配置

应用实例导航——Linux 系统基本网络配置

※场景呈现

A 公司需要将内部局域网的所有主机都接入 Internet，局域网内设置了一台主机作为网关，它通过 ADSL 接入 Internet。局域网内的其他主机通过该网关接入 Internet。

要求分别对局域网内的主机进行相应的网络参数配置，并在网关主机上建立 ADSL 连接，使其连上 Internet。

※技术要领

(1) 了解 Linux 系统中与网络相关的配置文件。
(2) 配置局域网内主机的网络参数。
(3) 利用 re-pppoe 软件进行 ADSL 连接的配置。

本节讲述如何使 Linux 服务器连接 Internet。一般来说，不管是 PC 还是服务器，最终都是通过网络服务供应商(ISP)接入 Internet 的，现在最流行的方式是通过 ADSL 拨号连接 Internet。当网关服务器连接 Internet 后，其子网内的主机就可以通过配置相应的网络参数经网关连接 Internet。因此，本节将分别介绍这两种联网的方式，在此之前，还要简单介绍一下 Linux 系统下与网络相关的几个配置文件的功能和意义，这在后面的章节中也会经常用到。

2.2.1 Linux 网络相关配置文件简介

在 Linux 服务器连接 Internet 之前，需要首先介绍 Linux 系统中与网络相关的一些配置文件，Linux 需要通过读取这些网络配置文件中的参数才能连上 Internet。

- /etc/sysconfig/network
  该配置文件用于控制与网络有关的文件，及设置守护进程的行为参数。HOSTNAME、NETWORKING、GATEWAY 等重要的网络参数都在此设定。
- /etc/sysconfig/network-scrips/ifcfg-ethn
  该配置文件用于设定网卡的参数，ethn 表示网卡号，第一块网卡的号码为 eth0，第二块为 eth1，以此类推。该文件包含了网卡的 network、IP、broadcast、gateway 和 DHCP 等重要参数。
- /etc/resolv.conf
  该配置文件用于设定域名搜索顺序和 DNS 服务器的地址，每一行遵循格式“关键字 设定值 1 设定值 2……”，即一个关键字对应一个或多个设定值，之间以空格

符隔开。详细的关键字及其描述如表 2-2 所示。

- /etc/hosts

  该配置文件用于设定 IP 地址和主机名之间的对应关系，以利于主机快速从主机名中查找到 IP 地址。本机所有主机名与 IP 的对应关系以及子网主机名与 IP 对应关系必须写入此文件中，图 2-14 显示了/etc/hosts 的一个实例，其中的标注部分是本机的两个主机名与 IP 的对应关系，下面列出了内部子网的主机名与 IP 地址的对应关系。外部主机名和 IP 的对应可以不列在该文件中，因为可以通过 DNS 服务器查询。

表 2-2　resolv.conf 文件的关键字描述

关 键 字	描　述
nameserver	表示 DNS 服务器的 IP 地址，通常设置多个 nameserver，当第一个 nameserver 没有响应时，继续查询下一个 namesserver
domain	表示主机的域名
search	与 domain 关键字不能并存。search 指定域名的查询顺序，当查找没有域名的主机时，就由 search 设定值指定的域进行逐一查找
sortlist	设定将得到的域名的排序方式，排序依据为网络/子网掩码的序列对

```
[root@cgsp ~]# vi /etc/hosts
211.65.63.146    cgsp.job      cgsp     cgsp.local
10.1.0.1         master.job    master   master.local
10.1.0.1         node1.job     node1    node1.local
10.1.0.2         node2.job     node2    node2.local
10.1.0.3         node3.job     node3    node3.local
10.1.0.4         node4.job     node4    node4.local
10.1.0.5         node5.job     node5    node5.local
10.1.0.6         node6.job     node6    node6.local
10.1.0.7         node7.job     node7    node7.local
10.1.0.8         node8.job     node8    node8.local
10.1.0.9         node9.job     node9    node9.local
10.1.0.10        node10.job    node10   node10.local
```

图 2-14　/etc/hosts 文件示例

- /etc/nsswitch.conf

  该配置文件用于管理多个配置文件的查找顺序，它是由 Sun 公司设计开发的，规定了系统到哪里查询主机名，当存在多个配置文件时规定以何种顺序去查询。每一行遵循格式“关键字：设定值 1 设定值 2……”，关键字后以冒号分隔，设定值之间以空格符分隔。详细的关键字及其描述如表 2-3 所示，设定值及其描述如表 2-4 所示。

表 2-3　nsswitch.conf 文件的关键字描述

关 键 字	描　述	关 键 字	描　述
aliases	邮件别名	protocols	网络协议
passwd	系统用户	services	服务名称和端口号
group	用户组	ethers	以太网号
shadow	用户的密码	rpc	远程过程调用的名称
hosts	主机名和 IP 地址	netgroup	网内组
networks	网络名称		

✧ /etc/services

该配置文件用于设定服务名称和端口号之间的对应关系，每一行遵循格式“服务名 端口号/协议名”，很多系统程序通过该文件查询系统服务使用了何种网络协议、端口号为多少。

点评与拓展：本节中介绍的这些与网络相关的配置文件，不仅在本章网络配置时有用，在后面章节配置服务器时也经常需要对这些文件进行配置，对这些文件的透彻理解会使后面章节的学习事半功倍。

表 2-4 nsswitch.conf 文件的设定值描述

设定值	描 述
files	除了 netgroup，对其他关键字都有效，到相应的/etc 目录下的配置文件去查询，比如 hosts 对应的配置文件就是/etc/hosts
db	除了 netgroup，对其他关键字都有效，到相应的/var/db 数据库中查询，数据库中的记录通常远大于配置文件中的记录数
compat	兼容模式，仅对 passwd、group 和 shadow 关键字有效
dns	只对 hosts 关键字有效，表示通过 DNS 服务器查询
nis	对所有关键字都有效，表示通过 NIS 服务器查询

2.2.2 通过 LAN 网关接入 Internet

很多公司机构和学术部门都采用通过 LAN 网关接入 Internet 的方式，网关利用 ADSL 接入 Internet。本节首先介绍单网卡的服务器如何通过网关接入 Internet，然后介绍多网卡的服务器如何既连接内部子网，又通过网关接入 Internet。

❶ 如果服务器只有一块网卡，其网卡号通常是 eth0，在确保网卡驱动安装无误、网卡正常工作的情况下，就可以对/etc/sysconfig/network-scrips/ifcfg-eth0 配置文件进行设定，利用 vi 命令打开该配置文件，如图 2-15 所示。进行如图 2-16 所示的设置，其中：DEVICE 设定项是网卡号，为 eth0；HWADDR 设定项是惟一标识该网卡的 MAC 地址；ONBOOT 设定项表示是否存在内部子网，单网卡主机通常设置为 no；IPADDR 设定项是该网卡所绑定的 IP 地址，这需要根据实际网络而定，通常利用网关连上 Internet 的机构或部门会规定 IP 地址的范围，该设定项可以在 IP 范围内任意选择，但是不能跟已经联网的 IP 地址冲突，本例中为 172.18.12.178；NETMASK 设定项是子网掩码，也由机构或部门规定，本例为 255.255.255.128；GATEWAY 设定项为网关的地址，就是直接与 Internet 相连主机的内部地址，也由机构或部门规定，本例为 172.18.12.129；TYPE 设定项为局域网类型，一般是以太网；IPV6INIT 设定项表示是否允许 IPv6 报文，一般设为 no。设定完毕后，可使用下面命令启动网络服务：

```
/etc/init.d/network start
```

如果网络服务已经启动，需要使用下面命令重启网络服务以重新读取配置文件的设置:

```
/etc/init.d/network restart
```

```
[root@seugrid2 network-scripts]# pwd
/etc/sysconfig/network-scripts
[root@seugrid2 network-scripts]# vi ifcfg-eth0
```

图 2-15　打开/etc/sysconfig/network-scrips/ifcfg-eth0 文件

```
# Intel Corporation 82801G (ICH7 Family) LAN Controller
DEVICE=eth0
BOOTPROTO=none
HWADDR=00:12:3F:C3:30:6B
ONBOOT=no
DHCP_HOSTNAME=seugrid2.seu.edu.cn
IPADDR=172.18.12.178
NETMASK=255.255.255.128
GATEWAY=172.18.12.129
TYPE=Ethernet
USERCTL=no
IPV6INIT=no
PEERDNS=yes
```

图 2-16　ifcfg-eth0 设定内容

Linux 提供了 ifconfig 系统命令查看所有网卡的信息，输入命令后的结果如图 2-17 所示。可以看到 shell 出现了两块网卡的信息，eth0 和 lo，其中的 eth0 即为上面设定的连接网关的网卡，lo 为环路网卡，绑定的是 127.0.0.1 本地 IP，仅用于测试。

```
[root@seugrid2 etc]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:12:3F:C3:30:6B
          inet addr:172.18.12.178  Bcast:172.18.12.255  Mask:255.255.255.128
          inet6 addr: 2001:da8:1002:9f8:212:3fff:fec3:306b/64 Scope:Global
          inet6 addr: fe80::212:3fff:fec3:306b/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3663528 errors:0 dropped:0 overruns:0 frame:0
          TX packets:93280 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:2908638657 (2.7 GiB)  TX bytes:7135998 (6.8 MiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:513432 errors:0 dropped:0 overruns:0 frame:0
          TX packets:513432 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:419954343 (400.4 MiB)  TX bytes:419954343 (400.4 MiB)
```

图 2-17　查看所有网卡的信息

❷ 接着以装有两块网卡的服务器为例讲述多网卡服务器的配置，两块网卡分别为 eth0 和 eth1，将 eth0 号网卡用于连接内部子网，eth1 号网卡用于连接外部网关从而接入 Internet。打开/etc/sysconfig/network-scrips/ifcfg-eth0 文件，将 IPADDR 设定项置为内部子网的 IP 地址，本例为 10.1.0.1；NETMASK 设定项置为内部子网的掩码，本例为 255.255.255.0；由于此时存在内部子网，因此 ONBOOT 设定项置为 yes；HWADDR 设为 eth0 号网卡的 MAC 地址，详细设定如图 2-18 所示。然后，打开/etc/sysconfig/network-scrips/ifcfg-eth1 文件，将 IPADDR 设定项置为该服务器对外的 IP 地址，本例为 211.65.63.146；HWADDR 设为 eth1 号网卡的 MAC 地址；ONBOOT 设定项同样置为 yes；此时必须设定 GATEWAY 地址，本例为 211.65.63.1，详细设定如图 2-19 所示。设定完毕后，同样需要启动或重启系统网络服务。同样可以使用 ifconfig 系统命令查看所有网卡的信息，输入命令后的结果如图 2-20 所示，

可以看到 eth0 和 eth1 两块网卡的信息都显示在了 shell 上,从中可以检查网卡的设定是否正确。

点评与拓展: 网卡设置时的 IP 地址、网关地址和子网掩码等网络参数是在局域网设计之初就确定好的,用户可以从网络管理员处获得。

```
[root@cgsp network-scripts]# vi ifcfg-eth0
BOOTPROTO=none
DEVICE=eth0
ONBOOT=yes
IPADDR=10.1.0.1
NETMASK=255.255.255.0
HWADDR=00:13:72:4F:80:72
TYPE=Ethernet
```

图 2-18　内部子网网卡的设置

```
[root@cgsp network-scripts]# vi ifcfg-eth1
DEVICE=eth1
ONBOOT=yes
BOOTPROTO=none
IPADDR=211.65.63.146
HWADDR=00:13:72:4F:80:73
TYPE=Ethernet
USERCTL=no
PEERDNS=yes
GATEWAY=211.65.63.1
IPV6INIT=no
```

图 2-19　连接外部网关网卡的设置

```
[root@cgsp ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:13:72:4F:80:72
          inet addr:10.1.0.1  Bcast:10.1.0.255  Mask:255.255.255.0
          inet6 addr: fe80::213:72ff:fe4f:8072/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:97252 errors:0 dropped:0 overruns:0 frame:0
          TX packets:89580 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:9229746 (8.8 MiB)  TX bytes:6580817 (6.2 MiB)
          Base address:0xecc0 Memory:fe6e0000-fe700000

eth1      Link encap:Ethernet  HWaddr 00:13:72:4F:80:73
          inet addr:211.65.63.146  Bcast:211.65.63.255  Mask:255.255.255.0
          inet6 addr: fe80::213:72ff:fe4f:8073/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:5588695 errors:0 dropped:0 overruns:0 frame:0
          TX packets:91835 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:503033134 (479.7 MiB)  TX bytes:60305248 (57.5 MiB)
          Base address:0xdcc0 Memory:fe4e0000-fe500000
```

图 2-20　查看所有网卡的信息

2.2.3　通过 ADSL 接入 Internet

目前直接接入 Internet 最流行的方法就是通过 ADSL,在局域网内作为网关的主机通常就利用该方法接入 Internet,本节将阐述 Linux 服务器如何使用 ADSL。

❶ Linux 系统提供的 rp-pppoe 软件就是用来通过 ADSL 接入 Internet 的，Fedora 6 默认安装了 rp-pppoe 软件的 3.5 版本。可以使用下面的命令检查 rp-pppoe 是否已安装(如图 2-21 所示):

```
rpm -qa | grep ppp
```

如果系统尚未安装 rp-pppoe 软件，利用 Fedora 6 安装包上的 rpm 包进行安装。

```
[root@seugrid2 local]# rpm -qa | grep ppp
ppp-2.4.4-1
rp-pppoe-3.5-32.1
```

图 2-21　检查 rp-pppoe 是否安装

❷ 建立 ADSL 连接之前，需要对该服务器的网络进行重新配置，打开/etc/sysconfig/network 文件，此时该文件只需设置前两个参数 NETWORKING 和 HOSTNAME 即可。GATEWAY 和 GATEWAYDEV 这两个参数一定要确保是空值，否则 ADSL 将无法连接，如图 2-22 所示。对用于 ADSL 的网卡也可以简化其配置，在 ADSL 连接中起作用的只有 DEVICE 和 ONBOOT 两个参数，其他参数即使设置也不起作用，如图 2-23 所示。最后使用下面的命令重启 Linux 网络服务使 network 和 eth0 设置生效，如图 2-24 所示。

```
/etc/init.d/network restart
```

另外，还需要将用于 ADSL 连接的 eth0 号网卡停止，输入下面命令：

```
ifdown eth0
```

```
[root@sec sysconfig]# pwd
/etc/sysconfig
[root@sec sysconfig]# vi network
NETWORKING=yes
HOSTNAME=sec
GATEWAY=
GATEWAYDEV=
```

图 2-22　重新进行网络设置

```
[root@sec sysconfig]# cd network-scripts/
[root@sec network-scripts]# vi ifcfg-eth0
DEVICE=eth0
ONBOOT=yes
```

图 2-23　简化 eth0 号网卡设置

```
[root@sec network-scripts]# /etc/init.d/network restart
```

图 2-24　重启网络服务

❸ 对用于接入 ADSL 的网卡重新配置后，就可以新建 ADSL 连接了，可通过执行/usr/sbin/adsl-

setup 命令来实现。执行该命令后，shell 首先提示输入 ADSL 账号，这个账号由 ISP 提供，接着输入用于接入 ADSL 的网卡号，即上步所停止的那块网卡，如图 2-25 所示。

```
[root@seugrid2 ~]# adsl-setup
Welcome to the ADSL client setup.  First, I will run some checks on
your system to make sure the PPPoE client is installed properly...


LOGIN NAME

Enter your Login Name (default root): YOUR ADSL

INTERFACE

Enter the Ethernet interface connected to the ADSL modem
For Solaris, this is likely to be something like /dev/hme0.
For Linux, it will be ethX, where 'X' is a number.
(default eth0): eth0
```

图 2-25　输入 ADSL 账号

接着根据 shell 提示输入 DNS 主服务器及备用服务器的地址，可根据实际情况进行设置；然后输入ADSL 账号的密码; USERCTRL 部分选择是否允许一般系统用户启动或停止 ADSL 连接，根据实际需要选择 yes 或 no，如图 2-26 所示。

```
DNS

Please enter the IP address of your ISP's primary DNS server.
If your ISP claims that 'the server will provide dynamic DNS addresses',
enter 'server' (all lower-case) here.
If you just press enter, I will assume you know what you are
doing and not modify your DNS setup.
Enter the DNS information here: 202.119.24.12
Please enter the IP address of your ISP's secondary DNS server.
If you just press enter, I will assume there is only one DNS server.
Enter the secondary DNS server address here: 202.119.24.18

PASSWORD

Please enter your Password:
Please re-enter your Password:

USERCTRL

Please enter 'yes' (three letters, lower-case.) if you want to allow
normal user to start or stop DSL connection (default yes): yes
```

图 2-26　设置 DNS 地址和 ADSL 密码

接着选择此 ADSL 连接的防火墙类型，通常选择 0；然后选择是否在系统启动时就启动此 ADSL 连接，一般选择 yes，如图 2-27 所示。这样，一个 ADSL 就创建完成了，shell 将显示上述几步所设置的详细内容， 用户确认后，shell 提示 ADSL 创建成功，如图 2-28 所示。

```
FIREWALLING

Please choose the firewall rules to use.  Note that these rules are
very basic.  You are strongly encouraged to use a more sophisticated
firewall setup; however, these will provide basic security.  If you
are running any servers on your machine, you must choose 'NONE' and
set up firewalling yourself.  Otherwise, the firewall rules will deny
access to all standard servers like Web, e-mail, ftp, etc.  If you
are using SSH, the rules will block outgoing SSH connections which
allocate a privileged source port.

The firewall choices are:
0 - NONE: This script will not set any firewall rules.  You are responsible
          for ensuring the security of your machine.  You are STRONGLY
          recommended to use some kind of firewall rules.
1 - STANDALONE: Appropriate for a basic stand-alone web-surfing workstation
2 - MASQUERADE: Appropriate for a machine acting as an Internet gateway
                for a LAN
Choose a type of firewall (0-2): 0

Start this connection at boot time

Do you want to start this connection at boot time?
Please enter no or yes (default no):yes
```

图 2-27　设置 ADSL 防火墙

```
** Summary of what you entered **

Ethernet Interface: eth0
User name:          YOUR ADSL
Activate-on-demand: No
Primary DNS:        202.119.24.12
Secondary DNS:      202.119.24.18
Firewalling:        NONE
User Control:       yes
Accept these settings and adjust configuration files (y/n)? yes
Accept these settings and adjust configuration files (y/n)? y
Adjusting /etc/sysconfig/network-scripts/ifcfg-ppp0
Adjusting /etc/resolv.conf
  (But first backing it up to /etc/resolv.conf.bak)
Adjusting /etc/ppp/chap-secrets and /etc/ppp/pap-secrets
  (But first backing it up to /etc/ppp/chap-secrets.bak)
  (But first backing it up to /etc/ppp/pap-secrets.bak)


Congratulations, it should be all set up!

Type '/sbin/ifup ppp0' to bring up your xDSL link and '/sbin/ifdown ppp0'
to bring it down.
Type '/sbin/adsl-status /etc/sysconfig/network-scripts/ifcfg-ppp0'
to see the link status.

[root@seugrid2 ~]#
```

图 2-28　ADSL 连接创建成功

❹ ADSL 连接创建成功后，可以使用下面命令接入 Internet(如图 2-29 所示):

```
adsl-start
```

```
[root@sec network-scripts]# adsl-start
```

图 2-29　利用 adsl-start 进行连接

📖**点评与拓展：**使用 ADSL 连接 Internet 时，要确保不设置网关地址，另外用于 ADSL 的网卡需要停止。

2.3　Linux 路由的架设

应用实例导航——Linux 单网卡主机架设路由

※场景呈现

A 公司需要在一台单网卡的主机上架设路由，要求既与外部网关的 ADSL 接入 Internet，又与内部子网的主机进行连接。

※技术要领

(1)　单网卡绑定多个 IP 地址。
(2)　Linux 系统的路由配置。

从 2.1.3 节“Internet 的路由”的论述中知道，路由像桥梁一样将不同网段的主机连接起来，路由在每个不同网段内都有一个 IP 地址，每个 IP 地址就需要对应一块网卡。当某台服务器只有一块网卡时，能否作为路由器呢？即能否由一块物理网卡提供多块逻辑网卡的功能呢？本节将解释这个问题。

2.3.1　创建虚拟网卡

与不同的 IP 绑定需要不同的逻辑上的网卡，将这些逻辑上有意义的网卡称为虚拟网卡。虚拟网卡指的不是一个物理实体，可能多个虚拟网卡共用一块实体网卡，所以只有一块网卡的服务器可以通过建立多个虚拟网卡来与不同网段的 IP 绑定，从而实现路由功能。下面首先讲述如何创建虚拟网卡。

❶ 假设一台服务器有两块物理网卡，eth0 和 eth1，eth0 号网卡用于连接内部子网，IP 为 10.1.0.1；eth1 号网卡用于连接外部网络，IP 地址为 211.65.63.146，如图 2-30 所示。现在停止 eth0 号网卡，并新建一块虚拟网卡替代 eth0 的功能，此虚拟网卡通过 eth1 的物理数据口进行通信。使用下面两条命令可停止 eth0，并新建 eth1 的虚拟网卡(如图 2-31 所示，eth1:0 网卡与 eth1 具有相同的 MAC 地址)：

```
[root@cgsp network-scripts]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:13:72:4F:80:72
          inet addr:10.1.0.1  Bcast:10.1.0.255  Mask:255.255.255.0
          inet6 addr: fe80::213:72ff:fe4f:8072/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:99276 errors:0 dropped:0 overruns:0 frame:0
          TX packets:122777 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:9425278 (8.9 MiB)  TX bytes:8717439 (8.3 MiB)
          Base address:0xecc0 Memory:fe6e0000-fe700000

eth1      Link encap:Ethernet  HWaddr 00:13:72:4F:80:73
          inet addr:211.65.63.146  Bcast:211.65.63.255  Mask:255.255.255.0
          inet6 addr: fe80::213:72ff:fe4f:8073/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:5903139 errors:0 dropped:0 overruns:0 frame:0
          TX packets:97623 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:531598159 (506.9 MiB)  TX bytes:63393151 (60.4 MiB)
          Base address:0xdcc0 Memory:fe4e0000-fe500000
```

图 2-30　两块物理网卡的信息

```
[root@cgsp ~]# ifdown eth0
[root@cgsp ~]# ifconfig eth1:0 10.1.0.1 netmask 255.255.255.0 broadcast 10.1.0.255 up
[root@cgsp ~]# ifconfig
eth1      Link encap:Ethernet  HWaddr 00:13:72:4F:80:73
          inet addr:211.65.63.146  Bcast:211.65.63.255  Mask:255.255.255.0
          inet6 addr: fe80::213:72ff:fe4f:8073/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:5731185 errors:0 dropped:0 overruns:0 frame:0
          TX packets:93091 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:516131531 (492.2 MiB)  TX bytes:60843705 (58.0 MiB)
          Base address:0xdcc0 Memory:fe4e0000-fe500000

eth1:0    Link encap:Ethernet  HWaddr 00:13:72:4F:80:73
          inet addr:10.1.0.1  Bcast:10.1.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Base address:0xdcc0 Memory:fe4e0000-fe500000
```

图 2-31　创建 eth1:0

```
ifdown eth0
ifconfig eth1:0 10.1.0.1 netmask 255.255.255.0 broadcast 10.1.0.255 up
```

应注意上面第二条命令中的虚拟网卡的编号：eth1:0，eth1 表示用的是 eth1 号网卡的 MAC 地址，:0 表示该虚拟网卡的编号，第二块虚拟网卡是 eth1:1，第三块是 eth1:2，以此类推。命令接着设置了该虚拟网卡的属性，包括 IP 地址、子网掩码、广播地址等，最后的 up 参数表示启动该虚拟网卡，相当于命令：

```
ifup eth1:0
```

顺便总结一下，停止和启动网卡各有两种命令：

```
ifup(ifdown) ethi
ifconfig ethi up(down)
```

其中括号中的内容可作替换，ethi 泛指网卡号。

❷ 此时可以看一下创建 eth1:0 之后的路由表，输入 route 命令，结果如图 2-32 所示。第一处标注部分说明所有发送到 10.1.0.0 局域网的数据包通过 eth1 网卡接口传输，第二处标注部分说明所有发送到 211.65.63.0 外部网的数据包也通过 eth1 网卡接口传输，这不正是一块物理网卡绑定了两个 IP 地址，连接了两个不同的网段么？

```
[root@cgsp network-scripts]# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
10.1.0.0        *               255.255.255.0   U     0      0        0 eth1
192.168.10.0    *               255.255.255.0   U     0      0        0 ib1
211.65.63.0     *               255.255.255.0   U     0      0        0 eth1
default         211.65.63.1     0.0.0.0         UG    0      0        0 eth1
```

图 2-32　创建 eth1:0 后的路由表

可以通过 ping 虚拟网卡所绑定的 IP，测试虚拟网卡是否正常工作(如图 2-33 所示)：

```
ping 10.1.0.1
```

```
[root@cgsp ~]# ping -c 3 10.1.0.1
PING 10.1.0.1 (10.1.0.1) 56(84) bytes of data.
64 bytes from 10.1.0.1: icmp_seq=0 ttl=64 time=0.047 ms
64 bytes from 10.1.0.1: icmp_seq=1 ttl=64 time=0.024 ms
64 bytes from 10.1.0.1: icmp_seq=2 ttl=64 time=0.031 ms

--- 10.1.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2000ms
rtt min/avg/max/mdev = 0.024/0.034/0.047/0.009 ms, pipe 2
[root@cgsp ~]#
```

图 2-33　测试 eth1:0

❸ 2.2.1 节提到每块网卡在/etc/sysconfig/network-scripts 目录下都对应一个配置文件，名字为 ifcfg-ethn。为了方便虚拟网卡的启动和配置，需要为每个虚拟网卡建立一个 ifcfg-ethn:j 的配置文件，本例中由于 eth1:0 与 eth0 网卡配置文件的参数大致相同，因此可以直接复制 eth0 到配置文件，再稍作修改即可，如图 2-34 和 2-35 所示，将 DEVICE 改为 eth1:0，HWADDR 改为 eth1 的 MAC 地址，用到的命令分列于下：

```
cd /etc/sysconfig/network-scripts
cp ifcfg-eth0 ifcfg-eth1:0
vi ifcfg-eth1:0
```

```
[root@cgsp network-scripts]# cp ifcfg-eth0 ifcfg-eth1:0
```

图 2-34 复制 eth0 的配置文件并改名

```
BOOTPROTO=none
DEVICE=eth1:0
ONBOOT=yes
IPADDR=10.1.0.1
NETMASK=255.255.255.0
HWADDR=00:13:72:4F:80:73
TYPE=Ethernet
```

图 2-35 eth1:0 配置文件的参数

点评与拓展：值得注意的是，即便将虚拟网卡配置文件的 ONBOOT 设为 no，当 eth1 号网卡启动时，所有与 eth1 相关的虚拟网卡也将启动；而当 eth1 号网卡停止时，与其相关的虚拟网卡也将被停止。

2.3.2 Linux 路由的架设

成功创建虚拟网卡后，单网卡的服务器在逻辑上就有了多网卡，就可以直接开始配置路由了。一台具备路由功能的服务器需要有以下几个条件：

- ✧ 逻辑上多网卡连接不同的子网域，拥有不同的网域内的 IP 地址。
- ✧ 正确设定路由表使多个网域互联。
- ✧ 通过 ip_forward 开启 Linux 系统核心的路由功能。
- ✧ 不同网域内的客户机的网关设置为该路由服务器在该网域内的 IP。

❶ 有了上节创建虚拟网卡工作的铺垫，该服务器已经满足了前两个条件。接着就是开启 Linux 系统的核心路由功能，Linux 的核心程序都放在/proc 目录下，因此可使用下面的命令开启核心路由功能，如图 2-36 所示。

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

用 more 命令查看 ip_forward 文件，发现它只包含一个数字，0 或 1，1 表示开启路由功能。

```
[root@cgsp ~]# echo 1 > /proc/sys/net/ipv4/ip_forward
[root@cgsp ~]# more /proc/sys/net/ipv4/ip_forward
1
[root@cgsp ~]#
```

图 2-36 开启 Linux 核心路由功能

❷ 路由服务器配置完毕后，登录 10.1.0.0 网域内的某台客户机进行配置。以 10.1.0.2 为例，该

客户机的 eth0 号网卡用于连接 10.1.0.0 网域，使用 vi 命令编辑/etc/sysconfig/network-scripts/ifcfg-eth0 文件，将 GATEWAY 设为网关的地址：10.1.0.1，即上小节所创建的虚拟网卡所绑定的地址，如图 2-37 所示。使用下面命令重启网络服务使改动生效：

```
/etc/init.d/network restart
```

对 211.65.63.0 网段中的主机作同样的设置。在 10.1.0.2 客户机上 ping 211.65.63.0 网段中的主机测试路由服务器是否正常运行，使用下面命令测试：

```
ping 211.65.63.138
```

结果如图 2-38 所示，10.1.0.2 客户机可以 ping 通 211.65.63.138 客户机，说明路由服务器已经可以使不同网段的主机互联了。

```
DEVICE=eth0
BOOTPROTO=static
IPADDR=10.1.0.2
BROADCAST=10.1.0.255
NETWORK=10.1.0.0
NETMASK=255.255.255.0
ONBOOT=yes
GATEWAY=10.1.0.1
```

图 2-37　客户机网关的设置

```
[root@node2ib ~]# ping 211.65.63.138
PING 211.65.63.138 (211.65.63.138) 56(84) bytes of data.
64 bytes from 211.65.63.138: icmp_seq=0 ttl=128 time=0.502 ms
64 bytes from 211.65.63.138: icmp_seq=1 ttl=128 time=0.589 ms
64 bytes from 211.65.63.138: icmp_seq=2 ttl=128 time=0.521 ms
64 bytes from 211.65.63.138: icmp_seq=3 ttl=128 time=0.571 ms
64 bytes from 211.65.63.138: icmp_seq=4 ttl=128 time=0.512 ms

--- 211.65.63.138 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4003ms
rtt min/avg/max/mdev = 0.502/0.539/0.589/0.034 ms, pipe 2
```

图 2-38　测试路由服务器

点评与拓展： 由于本章中所涉及的配置经常需要停止或启动网卡，或者重启网络服务，这都将导致远程连接断开，因此建议在服务器前进行操作，而不是通过远程控制执行这些网络相关的命令。

2.4 本 章 小 结

本章回顾了 TCP/IP 协议族，重点介绍了 IP 地址和 MAC 地址、子网划分和子网掩码、网络层路由及 TCP、UDP 协议等内容，对网际互联、局域网组网技术等有深入理解的读者可以跳过本章第 2.1 节的学习，而对网络基本知识不甚了解的读者在阅读本章第 2.1 节的基础上，建议再阅读相关的网络教材。在 TCP/IP 协议族的基础上，本章讲述了 Linux 连上 Internet 的两种方法：通过局域网内的网关接入 Internet 和利用 ADSL 直接连接 Internet，并详细讲述了这两种方法的配置过程；最后讲述了如何在单网卡的服务器上架设路由，包括如何创建虚拟网卡以绑定多个 IP，如何进行 Linux 路由配置等实用技术。

第 3 章　Linux 防火墙与 NAT 服务

信息技术的快速发展使得以 Internet 为代表的计算机互联网络成为了现代社会最重要的信息基础设施之一。几乎所有的公司、政府和学术机构都将其设备和资料连接上了 Internet，但同时由此带来的网络安全问题越来越突出，也越来越受到人们的广泛重视。因此，如何有效防御恶意网络攻击成为架设 Linux 服务器时首先需要考虑的重要问题，其中应用最普遍最成熟的方法就是防火墙技术。防火墙技术试图将一切针对服务器的恶意网络攻击拒之门外，成为 Linux 服务器有效抵御恶意网络攻击的第一道坚实屏障。NAT(Network Address Translator，网络地址转换)服务是为解决 IP 地址即将耗尽的危机而提出的，连接内部局域网和外部互联网的 NAT 网关将内部私有 IP 地址转换成外部公有 IP 地址，这样，多个局域网就可以使用一个公共 IP 地址访问国际互联网，从而最终达到节省 IP 地址资源的目的。本章将详细介绍如何使用 iptables 防火墙软件来架设 Linux 包过滤防火墙，以及如何配置 NAT 服务以实现共享上网。

通过本章的学习，读者应掌握以下主要内容：

- ✧ 包过滤防火墙的基本原理
- ✧ iptables 的基本原理
- ✧ iptables 的基本配置
- ✧ 如何使用 iptables 架设 Linux 防火墙
- ✧ NAT 服务的基本功能
- ✧ 如何配置 NAT 网关

3.1　Linux 防火墙概述

3.1.1　防火墙简介

防火墙是一种有效的防御工具，当通过广域网和 Internet 访问内部的主机或者网络，以及通过内部的主机或者网络访问外部系统时，防火墙可以使系统免于受到网络攻击的威胁。

Bellare 早在 1994 年就列出了防火墙的几大目标：

- ✧ 所有的通信，无论是从外部到内部还是从内部到外部的，都必须经过防火墙。这一点可以通过阻塞所有未通过防火墙的对于本地网络的访问来实现。
- ✧ 只有被授权的通信才能通过防火墙，这些授权将在本地安全策略中规定。不同类型的防火墙实现不同的安全策略。

✧ 防火墙本身对于渗透必须是免疫的。这意味着必须使用运行安全操作系统的可信系统。

防火墙通常可以分为三类：包过滤防火墙、应用级网关和电路级网关。应用级网关也叫做代理服务器(Proxy Server)，在应用级通信中扮演着一个消息传递者的角色；电路级网关是一个独立系统，也可以由应用级网关在某个应用中执行。由于 Linux 系统自带的防火墙属于包过滤防火墙，因此重点介绍包过滤防火墙。

何谓包过滤防火墙呢？它是依据一套规则对收到的 IP 包进行处理，决定是转发还是丢弃，如图 3-1 所示。防火墙被设置成对两个方向(进入内部网络和从内部网络发出)的数据包进行过滤。过滤的具体处理方法根据数据包所包含的信息而定。

✧ 源 IP 地址：产生 IP 数据包的系统 IP 地址。
✧ 目的 IP 地址：IP 数据包所要到达的系统的 IP 地址。
✧ 源和目的传输层的地址：指数据包在源系统和目的系统经过的传输端口数，一些应用如 SNMP 和 TELNET 必须在这里进行。
✧ IP 协议域：对传输协议的定义。
✧ 接口：对那些有两到三个端口的路由器，这部分信息规定了哪个是数据包的进入端口，哪个是预留给数据包的端口。

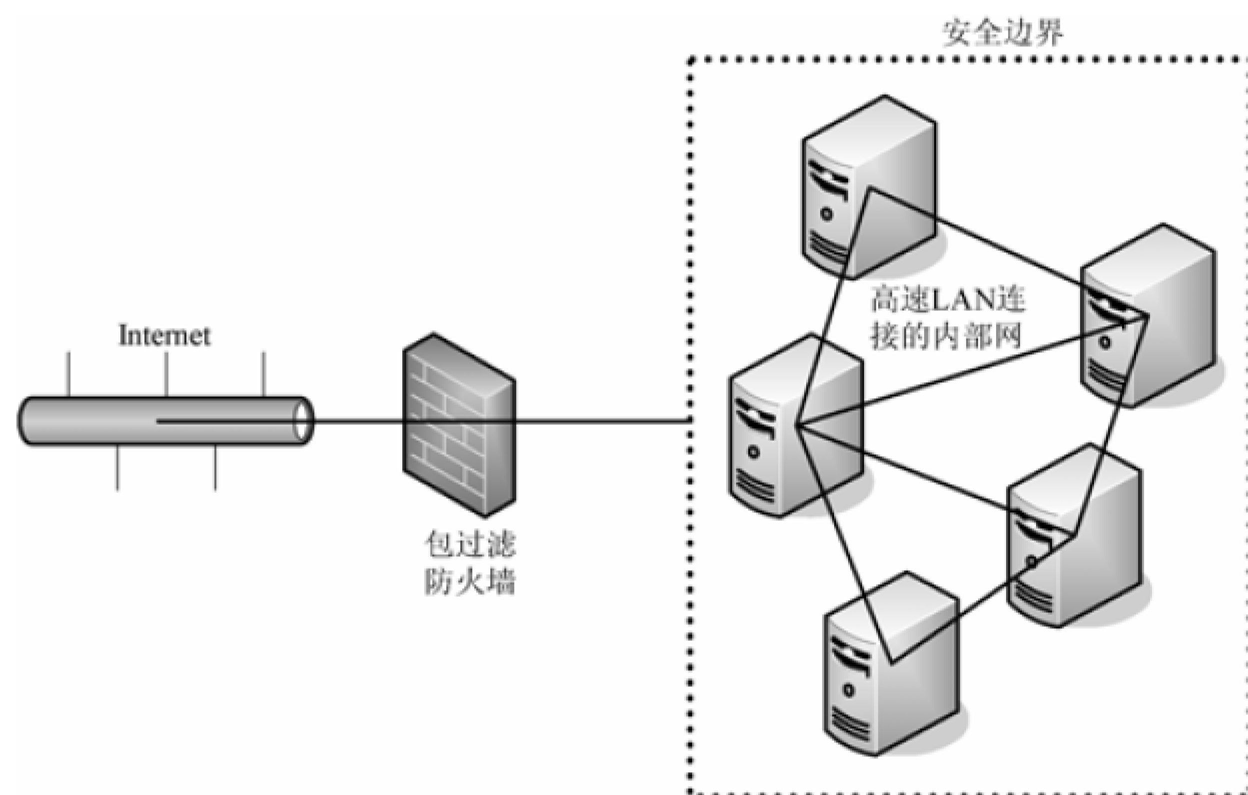


图 3-1　防火墙的部署位置

包过滤防火墙实际上是根据一个规则表，由规则表和 IP 报头或 TCP 报头内容的匹配情况来执行过滤操作的。如果有一条规则和数据包的状态匹配，就按照这条规则来执行过滤操作；如果没有一条规则匹配，就执行默认操作。默认的策略可能是以下两种。

✧ 默认丢弃：所有未明确允许转发的数据包都将被丢弃。
✧ 默认转发：所有未明确规定需要丢弃的数据包都将被转发。

默认丢弃策略显得比较保守。起初，所有的服务都会被阻塞，服务必须依靠实例的积累逐步地扩展，这时的防火墙在用户看来就像是一个障碍物。而默认转发的策略方便了用户的使用，但也相应降低了安全性；网络管理员基本上要对每一个被发现的安全威胁立刻做出反应。

包过滤防火墙的最大优点在于它的简单性，包过滤防火墙对用户而言几乎是透明的，

处理速度也很快。但是，包过滤防火墙也有着不容忽视的缺点，比如，包过滤防火墙不检查上层数据，对于那些利用特定应用程序的攻击，包过滤防火墙无法防范；包过滤防火墙容易受到利用 TCP/IP 规定和协议栈漏洞的攻击；无法支持高级用户认证方案等。因此，人们针对一些包过滤防火墙经常遭受的攻击的情况总结出了相应的一些对策，尽管这些对策不能完全防范各种各样的网络攻击，但是却也能看得出人们在防范网络攻击上经验的积累；同时，针对网络攻击制定相应的对策也是一个开放的话题，简洁有效的对策终将被人们所采纳。我们将目前的一些攻击和对策列举如下。

- ✧ IP 地址欺骗：入侵者从防火墙外部发送一个源地址为内部主机的数据包，试图利用假的地址来进入那些仅对源地址信赖的系统，在这些系统里，一旦数据包的源地址为防火墙内部的可信主机，它就将被允许通过。应对这种攻击的方法是，一旦在防火墙的外部接口处发现源地址是内部地址的数据包，就将它丢弃。
- ✧ 源路由攻击：攻击者在来源位置注明数据包在 Internet 上传输时所应该采用的路由，由此希望绕过存在的安全措施。应对策略是丢弃所有使用了这个选项的数据包。
- ✧ 微分片攻击：入侵者使用 IP 分片选项来制造出非常小的分片，分片如此之小，使得 TCP 头信息只能被放在一个独立的分片中。攻击者试图让包过滤防火墙仅仅检查第一个分片，然后将后面的所有分片全部放行。这种攻击方法用来对付那些过滤规则只能依赖于 TCP 头信息的防火墙是很有效的。然而，如果防火墙将那些协议类型为 TCP、IP 碎片偏移为 1 的小分片都丢弃，这种攻击也就失效了。

3.1.2 iptables 简介

iptables 的前身是 ipchains，ipchains 随 Linux 2.2.x 内核一起发布，当 Linux 内核发展到 2.4.x 后，ipchains 逐渐被功能更胜一筹的 iptables 所替代。与以前的 ipchains 和 ipfwadm 防火墙相比，iptables 最大的优点是它可以配置有状态的防火墙，即 iptables 能检查数据包的源和目的 IP 地址、源和目的端口、流入数据包的顺序号、TCP 分片的先后顺序以及头标记(SYN、ACK、FIN、RST 等)的状态，并且 iptables 可以跟踪整个连接会话，从而使整个过滤过程相互关联，这可以提高数据包的处理效率和响应速度。iptables 包过滤防火墙由两个组件构成，一个是 netfilter，另一个是 iptables，下面对这两个组件的功能和作用做简单的介绍。

(1) netfilter 组件

netfilter 组件也叫做内核空间，被集成在 Linux 的内核之中。netfilter 是一种内核中用于扩展各种网络服务的结构化底层框架，netfilter 的设计思想是生成一个模块结构使之能够比较容易的扩展，新的属性添加到内核中就不需要重新启动内核。这样，可以通过简单地构造一个内核模块来实现网络新特性的扩展，给底层的网络特性扩展带来了极大的便利，使更多从事网络底层研发的开发人员能够集中精力实现新的网络特性。

netfilter 组件主要由数据包过滤表(tables)组成，其中包含了控制 IP 包处理的规则集(rules)。根据规则所处理的 IP 包的类型，规则被分组放在链(chains)中，从而使内核对来自某些源、前往某些目的地或具有某些协议类型的数据包进行相应的处理，如完成数据包的处理、控制和过滤等工作。

(2)　iptables 组件

iptables 组件也叫做用户空间，用户通过它来插入、删除和修改规则链中的规则，并将这些规则告诉内核中的 netfilter 组件。

在使用 iptables 之前，读者首先需要理解规则、链和表这 3 个概念，这 3 个概念贯穿了 iptables 配置和应用的整个流程。

✧ 规则(rules)：是网络管理员预定义的条件，规则一般的定义为“如果数据包符合这个条件，就这样处理数据包”。规则存储在内核空间的信息包过滤中，这些规则分别指定源地址、目的地址、传输协议类型(如 TCP、UDP 和 ICMP)和服务类型(HTTPs、SMTP 和 FTP)。当数据包与所制定的规则相符合时，iptables 就根据规则所定义的处理方法来处理这个数据包，如接受(ACCEPT)、转发(FORWARD)或丢弃(DROP)等。

✧ 链(chains)：是数据包的传播路径，每一条链就是一系列规则检查清单，是规则的集合。当一个数据包到达一个链时，iptables 从链中的第一条规则开始检查该数据包是否符合规则所定义的条件。如果符合，iptables 就根据这条规则所定义的方法处理该数据包；如果不符合，iptables 继续检查下一条规则。如果该数据包不符合链中的所有规则，iptables 则根据该链预定义的默认策略来处理这个数据包。

✧ 表(tables)：提供特定的功能，iptables 包含 3 张表：filter 表、nat 表和 mangle 表，分别实现包过滤、网络地址转换和包重构的功能。filter 表主要用于过滤数据包，该表根据预定义的一组规则过滤符合条件的数据包；nat 表用于网络地址转换，可以实现共享上网功能；mangle 表用于对指定的包进行修改，因为某些特殊应用可能需要改写数据包的一些传输特性，不过在实际中该表的使用率不高。

3.1.3　iptables 的数据传输流程

iptables 通过由一系列规则定义的链完成数据包的传输，图 3-2 描述了 iptables 的数据传输流程。

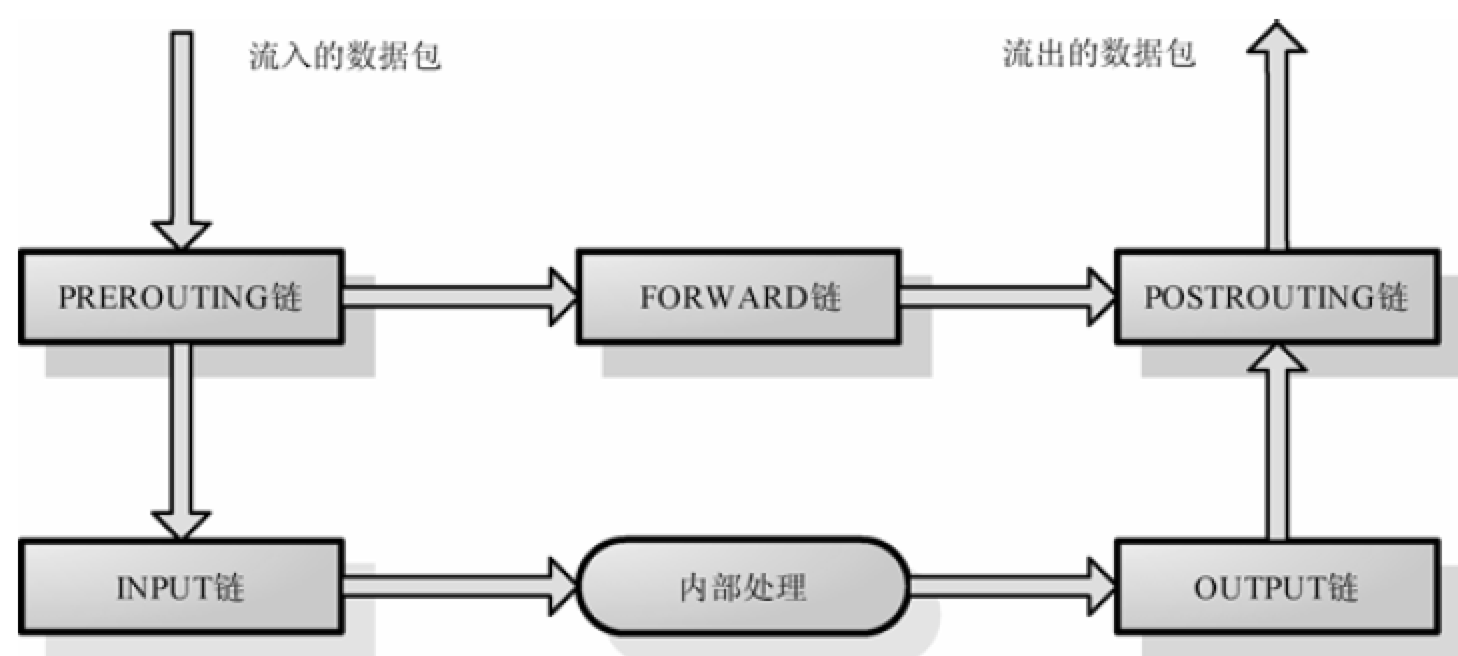


图 3-2　iptables 的数据传输流程

当一个数据包穿过防火墙流入时，它首先进入 iptables 的 PREROUTING 链，PREROUTING 链根据该数据包的目的 IP 地址判断是否需要将此数据包转发出去。如果数

据包的目的 IP 地址就是本机，那么该数据包将沿着图中向下箭头指示的方向移动，到达 INPUT 链。这时，主机上的所有进程都会收到这个数据包，进而根据该数据包头的其他信息决定由哪个进程来对数据包进行处理。如果数据包的目的 IP 地址不是本机 IP，那么防火墙就要将该数据包转发出去。数据包沿着图中向右箭头指示的方向移动，经过 FORWARD 链，到达 POSTROUTING 链流出。当本机进程需要发送数据包时，这些数据包就经过 OUTPUT 链，再经由 POSTROUTING 链输出。

数据包进入图中每一个链时，iptables 都会从链中第一条规则开始对照来处理数据包。数据包头的信息如果与链中某条规则所规定的条件完全匹配，就对该数据包执行该规则所定义的操作；数据包头的信息如果与链中的所有规则都不匹配，则将根据该链的默认策略来决定如何处理。一般的默认策略是丢弃该数据包。

链的规则及默认策略是由网络管理员在配置防火墙时指定，因此，配置 Linux 包过滤防火墙就是依照 iptables 的数据传输流程图为流程中的每条链添加规则的过程。

3.2 iptables 的基本配置

应用实例导航——iptables 过滤表的配置

※场景呈现

为 A 公司配置 Linux 防火墙，通过对 iptables 的 filter 表的配置实现默认策略的制定，实现对 TCP、UDP 数据包根据各种条件(如源、目标 IP 地址，源、目标端口号等)进行过滤的功能，并对某些类型的 ICMP 数据包进行控制。

※技术要领

(1) 制定 Linux 防火墙的默认策略。
(2) 利用 filter 表过滤 TCP、UDP 报文。
(3) 控制 ICMP 报文。

从概述部分知道，目前的 Linux 防火墙都通过 iptables 组件来实现，因此架设 Linux 防火墙是通过配置 iptables 组件来实现的。本节介绍如何利用 iptables 中的 Filter 表实现 Linux 防火墙默认策略的制定、TCP/UDP 数据包的过滤以及 ICMP 报文的控制等，最后讲述如何启动和关闭 Linux 防火墙。

3.2.1 iptables 策略的配置

iptables 内置了 3 张表：Filter 表、NAT 表和 Mangle 表，分别实现包过滤、网络地址转换和包重构的功能。

❶ iptables 的 3 张表都包含了多条链，每条链又包含了多条规则，规则是 Linux 防火墙的最小元素。那么如何查看 iptables 表的规则呢？可以使用下面的 shell 命令格式：

```
iptables [-t 表名] <-L> [-nv]
```

参数说明如下。

✧ -t：定义查看哪个表的规则列表，后面跟的表名可以是 filter、nat 和 mangle，如果此处不指定表名，则 iptables 默认查看 Filter 表。
✧ -L：列出目前表或链的规则。
✧ -n：不进行 IP 地址与主机名的反查，加快响应速度。
✧ -v：列出更多的信息，包括通过该规则的 IP 包的总位数，及相关的网卡接口。

图 3-3 列出了 NAT 表所有链的规则，图中的 Chain PREROUTING、Chain POSTROUTING、Chain OUTPUT 分别表示了 NAT 表中的三根链，在 3.5 节中将详细介绍。policy ACCEPT 表明当前 NAT 表所有链的默认策略都是 ACCEPT。

```
[root@seugrid3 ~]# iptables -t nat -L -n
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
[root@seugrid3 ~]#
```

图 3-3　查看 iptables 的 NAT 表规则

❷ 接着再查看一下 Filer 表的链及其规则，如图 3-4 所示，可以看到 Filter 表共有三条链：INPUT、FORWARD 和 OUTPUT。其中 INPUT 链中有 3 条规则，第 1 条规则说明 Linux 防火墙接受 TCP 端口从 5901 到 5905 的 IP 数据包，第 2 条规则说明 Linux 防火墙接受 TCP 端口为 5901 的 IP 数据包，而第 3 条规则说明 Linux 防火墙拒绝了 TCP 端口为 5901 的 IP 数据包。在这种互相矛盾的情况下，Linux 防火墙采取的策略是优先满足最新添加的规则，即命令行靠近最上面的规则(这里就是前述第 1 条规则)。如图 3-4 中所示命令，Linux 防火墙将接受 TCP 端口为 5901 的所有 IP 数据包。

```
[root@seugrid3 ~]# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpts:5901:
5905
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:5901
DROP       tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:5901

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

图 3-4　查看 iptables 的 Filter 表规则

❸ 在 3.1.1 节“防火墙简介”中讲述过什么是防火墙的默认策略，有默认丢弃和默认转发两种策略。策略对于 Linux 防火墙来说非常重要，它规定了防火墙对于那些 IP 数据包的头信息

不在用户设定的规则之内时，如何处置该 IP 数据包的动作，即规定了防火墙对所有 IP 数据包的默认动作。如果需要更改 iptables 的策略，则利用下面的命令格式完成：

```
iptables [-t 表名] -P[INPUT,OUTPUT,FORWARD] [ACCEPT,DROP]
```

参数说明如下。

✧ -P：说明该命令是定义策略的，后面跟表里面的链名。值得注意的是，这个 P 为大写，Linux 系统的命令行是区分大小写的。

✧ ACCEPT：所有未明确规定需要丢弃的数据包都将被转发。

✧ DROP：所有未明确允许转发的数据包都将被丢弃。

图 3-5 中，将 Filter 表 INPUT 链的默认策略改成了 DROP；其他两条链 FORWARD 和 OUTPUT 的默认策略被设为 ACCEPT。

```
[root@seugrid3 ~]# iptables -I INPUT -p tcp --dport 22 -j ACCEPT
[root@seugrid3 ~]# iptables -P INPUT DROP
[root@seugrid3 ~]# iptables -L -n
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:22
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpts:5901:5905
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:5901
DROP       tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:5901

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
[root@seugrid3 ~]#
```

图 3-5　更改 iptables 的策略

点评与拓展：不失一般性，INPUT 链是用于规定接受外部主机 IP 数据包的处理方式的，所以一般将其策略定为 DROP，这属于严格的策略。而 FORWARD 和 OUTPUT 两条链是用于处理内部子网的 IP 数据包的，由于网关对内部子网的 IP 数据包比较信任，所以将其策略定为 ACCEPT 相对比较宽松些。

3.2.2　添加 TCP/UDP 数据包的规则

在 2.1 节“TCP/IP 协议族概述”中介绍过，TCP 和 UDP 都是封装了网络层的 IP 包，它们在实现方面的主要区别在于 TCP 和 UDP 所采用的传输协议不同；TCP 和 UDP 都默认地指定了一部分端口，这些端口是不可更改的；另外一些未指定的端口既可以用作 TCP 端口，又可以用作 UDP 端口，因此，当打开或关闭端口时，通常需要指明是 TCP 还是 UDP 的端口。

❶ 图 3-6 演示了开放端口号为 22 的 TCP 端口的命令及运行结果。命令格式为：

```
iptables [-AI 链名] [-io 网卡号] [-p 协议类型] [-s 源 IP 网域] [-d 目的 IP 网域] -j [ACCEPT | DROP]
```

```
[root@seugrid3 ~]# iptables -I INPUT -p tcp --dport 22 -j ACCEPT
[root@seugrid3 ~]# iptables -P INPUT DROP
[root@seugrid3 ~]# iptables -L -n
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:22
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpts:5901:5905
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:5901
DROP       tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:5901

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
[root@seugrid3 ~]#
```

图 3-6　开放 TCP 22 号的端口

参数说明如下。

- --AI 链名：针对某些链进行规则的插入或累加。
- -A：新增加一条规则，且该规则被加在已有规则列表的最后面。需要注意的是，这个参数不使用规则编号。
- -I：插入一条新规则。如果没有指定此规则的顺序，它就被默认成为第一条规则。这个参数比较重要，由于 Linux 防火墙优先执行前面的规则，因此指定规则顺序很必要。比如，假设已有四条规则，其编号依次为 1～4，利用-I 参数插入一条新规则后，则这条新规则的编号被编为 1，原来的四条规则则按原来顺序其编号依次增 1，结果变为 2～5。
- --io：规定网卡编号，i 指定输入网卡号，o 指定输出网卡号。当服务器只有一块网卡时，可以不指定这个参数。
- -i 为 IP 数据包从其进入的网卡编号，多是指由外部 IP 地址主机所发的包其经过的网卡号；
- -o 为 IP 数据包从其出去的网卡编号，一般是内部子网所发的包其经过的网卡号。
- -p：规定 IP 数据包的协议格式，可以是 tcp、udp、icmp 和 all。
- -s：源 IP 网域，规定 IP 数据包的来源地址，可以指定是单独的 IP 地址，当然也可以是网域名，二者皆可。
- -d：目的 IP 网域，规定 IP 数据包的目的地址，可以指定是单独的 IP 地址，也可以是网域名，二者皆可。
- -j：选择怎样处理数据包，如接受或丢弃等。

❷ 图 3-7 演示了 iptables 规则的几种设定方法，图中的第一条命令表示只要是网卡号为 eth0 的 IP 数据包全部都接受；第二条命令表示只要是 IP 地址为 172.18.12.225 发出的数据包全部都接受；第三条命令表示源 IP 地址为子网 192.168.1.10 的数据包全部丢弃，为网域 192.168.1.0/24 的数据包全部接受。图 3-8 中的标示部分，显示了上述命令的结果全都写入了 ipfliter 表的 INPUT 链中。

```
[root@seugrid3 ~]# iptables -A INPUT -i eth0 -j ACCEPT
[root@seugrid3 ~]# iptables -A INPUT -s 172.18.12.225 -j ACCEPT
[root@seugrid3 ~]#
[root@seugrid3 ~]# iptables -A INPUT -s 192.168.1.10 -j DROP
[root@seugrid3 ~]# iptables -A INPUT -s 192.168.1.0/24 -j ACCEPT
[root@seugrid3 ~]#
```

图 3-7　iptables 规则的几种设定方法

```
[root@seugrid3 ~]# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0           icmp type 8
DROP       udp  --  0.0.0.0/0            0.0.0.0/0           udp dpt:21
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:22
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpts:5901:5905
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:5901
DROP       tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:5901
DROP       icmp --  0.0.0.0/0            0.0.0.0/0           icmp type 8
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0           icmp type 8
DROP       icmp --  0.0.0.0/0            0.0.0.0/0           icmp type 8
DROP       icmp --  0.0.0.0/0            0.0.0.0/0           icmp type 8
DROP       icmp --  0.0.0.0/0            0.0.0.0/0           icmp type 8
DROP       icmp --  0.0.0.0/0            0.0.0.0/0           icmp type 8
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     all  --  172.18.12.225        0.0.0.0/0
DROP       all  --  192.168.1.10         0.0.0.0/0
ACCEPT     all  --  192.168.1.0/24       0.0.0.0/0

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
[root@seugrid3 ~]#
```

图 3-8　设定后的结果

3.2.3　添加 ICMP 数据包的规则

ICMP 协议用于网络链路的连通性检测和故障消息传递，其类型相当多。ICMP 协议是个无连接协议，只要封装好 ICMP 报文并发送，就可以从网络上传递到目的地址主机。这使得非常易于伪造 ICMP 报文，因而 ICMP 协议常被用于发起 DoS 攻击。Linux 防火墙则可以对各种类型的 ICMP 数据包进行灵活的控制。

❶ 图 3-9 演示了关闭类型 8 的 ICMP 包的命令及运行结果。其命令格式如下：

```
Iptables -A INPUT -p icmp --icmp -type 类型 -j [ACCEPT | DROP]
```

```
[root@seugrid3 ~]# iptables -A INPUT -p icmp --icmp-type 8 -j DROP
[root@seugrid3 ~]#
```

图 3-9　关闭类型 8 的 ICMP 包

❷ 类型 8 的 ICMP 用于响应远方主机，一旦被关闭后，将不响应远方主机的询问。读者可以用 ping 命令测试类型 8 的 ICMP 关闭前后的应答情况。

3.2.4　Linux 防火墙的开启与关闭

安装好 Linux 系统后，防火墙默认开启，这时防火墙经常会跟用户的设置起冲突，因此用户在学习 iptables 之前，可以先关闭防火墙，这样能方便其他的配置与测试。

❶ 执行 setup 命令启动图形化工具菜单，弹出如图 3-10 所示的界面。

❷ 选择【选择一种工具】|【防火墙配置】菜单命令进入防火墙设置界面，如图 3-11 所示。单击空格键选择【启用】或【禁用】命令来启用或禁用防火墙，按 Tab 键切换界面中的选项。

❸ 当防火墙处于启用状态时，选择【防火墙配置】|【定制】菜单命令可进入防火墙定制界面，如图 3-12 所示。第一行选择信任的网卡号，通常选择连接内部局域网的网卡号，这样来自内部局域网的 IP 数据包都能接受；第二行选择需要 IP 伪装的网卡号，即将在 3.4 节中讲述的 NAT 技术；第三行选择需要开启的服务，相当于添加一条防火墙规则；最后一行，开放

其他端口，比如有时 Web 服务器端口采用 9080 时，WWW 服务默认端口是 80 端口，因此需要在这里额外开放 9080 端口。



图 3-10　图形化工具菜单

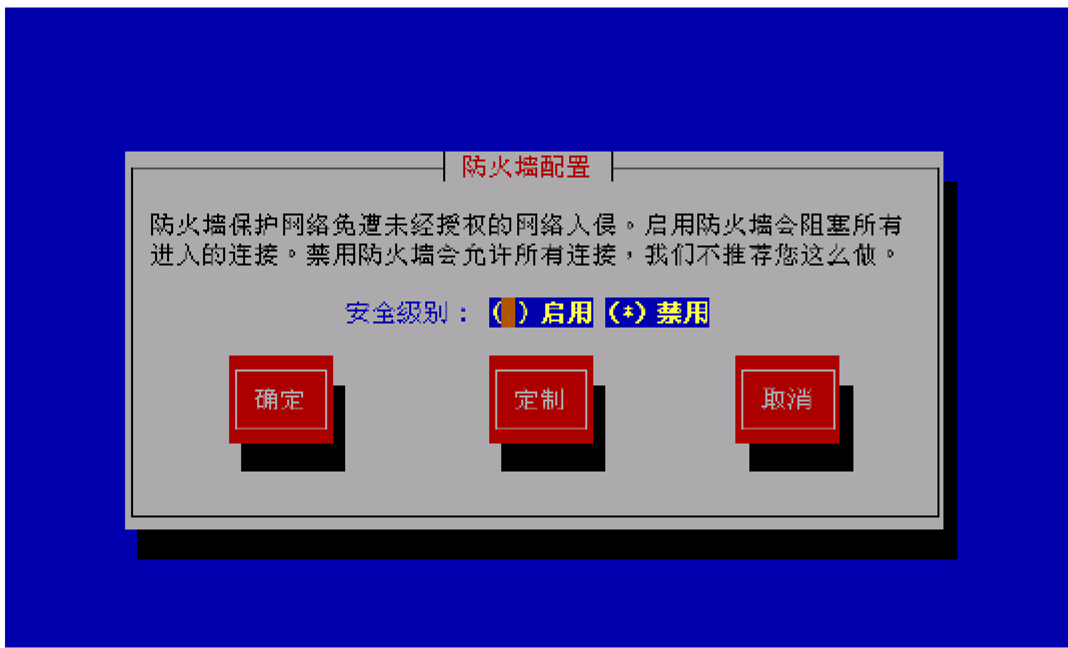


图 3-11　防火墙启动菜单



图 3-12　防火墙定制菜单

📖**点评与拓展**：值得注意的是 TCP/UDP 和 ICMP 的区别，添加 TCP/UDP 相关规则时需要指定端口号，而添加 ICMP 规则时需要指定类型。

3.3　架设 Linux 防火墙

应用实例导航——架设实用 Linux 防火墙

※场景呈现

A 公司内的所有主机都是通过一个网关连上 Internet，访问外部资源。内部主机都通过高速 LAN 形成内部子网，网络拓扑如图 3-13 所示。出于网络安全性考虑，现在该公司计划

在网关上架设一道防火墙，对从 Internet 流入公司内部局域网的 IP 数据包加以严格控制；而对公司局域网内部的主机则比较信任，因此只要对内部主机流出的IP数据包稍加控制即可。

防火墙的基本要求及假设条件如下。

(1) 网关上配置两块网卡：外部连接 Internet 的网卡号为 eth1，内部连接高速 LAN 的网卡号为 eth0。

(2) 网关需要开放 www、SSH、SMTP 等服务。

要求作为网络管理员的您写出该 Linux 防火墙的配置脚本，并且要求这个脚本稍做修改就可以用于其他网络拓扑的情形，即具有普适性；最后要求详细描述该 Linux 防火墙的工作流程。

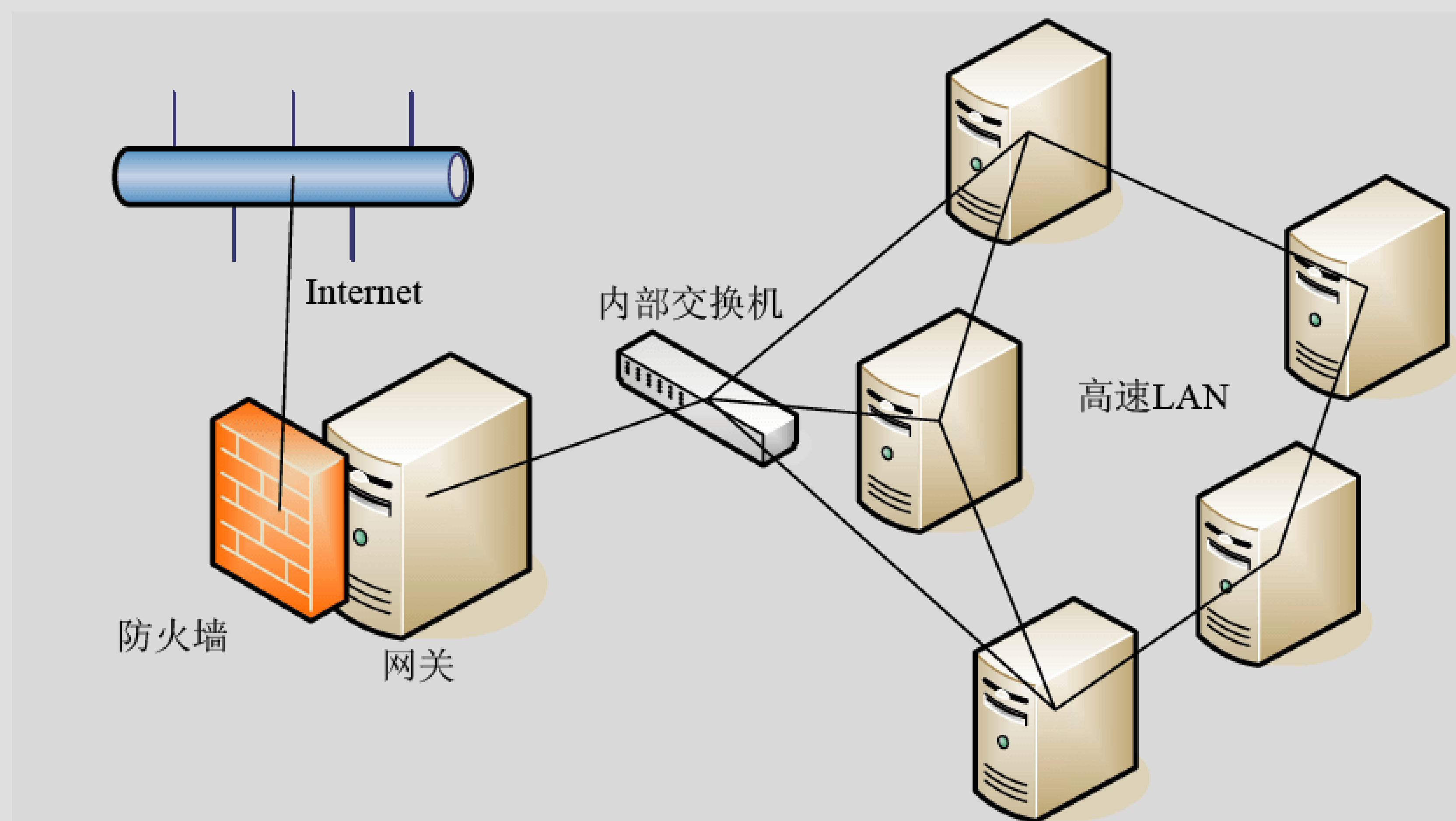


图 3-13　某公司的网络拓扑图

※技术要领

(1) 网卡的基本配置脚本。
(2) 对外默认策略、开启服务规则的指定。
(3) 对内策略、链规则的指定。
(4) Linux 简单 shell 脚本的书写和执行，并需要有一定的普适性。

下面将详细讲述如何架设这个 Linux 防火墙。通常，当配置一些复杂软件的时候，网络管理员都不希望通过一行一行地输入 shell 命令来完成。因为，这样不仅效率比较低，而且一旦中间发生错误，就不得不需要又从头开始一条一条地重新输入，所以，人们一般采用 shell 脚本的方式来完成这种配置。下面分三块来分别介绍这个配置脚本的内容，这三块的配置内容分别是：防火墙网络的基本配置、连接 Internet 外网的配置和连接高速 LAN 子网的配置。

❶ 如图 3-14 所示，首先使用 vi 命令建立脚本文件 rules。#!/bin/bash 是 shell 脚本文件的首行，它规定了该文件是一个脚本文件。在网络的基本配置方面，定义了三个变量：EXTIF，定义

了连接外部 Internet 的网卡号；INIF，定义了连接内部局域网的网卡号，如果这台服务器只有一块网卡，那么 INIF= “ ”；INNET，定义了内部局域网的子网号，若不存在子网，则 INNET = “ ”。

```
[root@seugrid3 iptables]# vi rules
#!/bin/bash
#!/bin/bash
  EXTIF="eth1"              # Extern IP
  INIF="eth0"               # LAN; if none ""
  INNET="192.168.1.0/24"    # LAN subnetwork, if none ""
  export EXTIF INIF INNET
```

图 3-14 脚本的基本网络配置部分

❷ 如图 3-15 和图 3-16 所示，演示了连接 Internet 外网的配置，一共分为 5 个部分(见图中脚本＃1～＃5 的注释标号)。第 1 部分，设定了核心的网络功能，这一部分基本不需要改动就能适用于其他情形；第 2 部分，清除了 iptables 已有的规则，并定义默认策略，以及设定与网卡有关的选项；第 3 部分，启动额外的防火墙 script 模组，也不需改动就能适用于其他情形；第 4 部分，设置 ICMP 相关的参数，脚本文件将允许进入的 ICMP 包类型赋给了参数 AICMP，可以根据实际需求添加 ICMP 类型，然后利用循环语句逐一开通；第 5 部分，开通允许进入的 TCP/UDP 服务，为了满足防火墙假设条件的要求，这里开通了 SSH、SMTP、DNS、WWW、POP3 和 HTTPS 等服务，该部分可以根据实际需求添加相关脚本语句。

```
# Extern IP
# 1.key basic function:
  echo "1" > /proc/sys/net/ipv4/tcp_syncookies
  echo "1" > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
  for i in /proc/sys/net/ipv4/conf/*/rp_filter; do
        echo "1" > $i
  done
  for i in /proc/sys/net/ipv4/conf/*/log_martians; do
        echo "1" > $i
  done
  for i in /proc/sys/net/ipv4/conf/*/accept_source_route; do
        echo "0" > $i
  done
  for i in /proc/sys/net/ipv4/conf/*/accept_redirects; do
        echo "0" > $i
  done
  for i in /proc/sys/net/ipv4/conf/*/send_redirects; do
        echo "0" > $i
  done

# 2. Clear rules; Specify policy.
  PATH=/sbin:/usr/sbin:/bin:/usr/bin; export PATH
  iptables -F
  iptables -X
  iptables -Z
  iptables -P INPUT   DROP
  iptables -P OUTPUT  ACCEPT
  iptables -P FORWARD ACCEPT
  iptables -A INPUT -i lo -j ACCEPT
  iptables -A INPUT -m state --state RELATED -j ACCEPT
```

图 3-15 连接 Internet 外网的配置部分(#1～#2)

```
# 3. external script group
  if [ -f /usr/local/virus/iptables/iptables.deny ]; then
        sh /usr/local/virus/iptables/iptables.deny
  fi
  if [ -f /usr/local/virus/iptables/iptables.allow ]; then
        sh /usr/local/virus/iptables/iptables.allow
  fi
  if [ -f /usr/local/virus/httpd-err/iptables.http ]; then
        sh /usr/local/virus/httpd-err/iptables.http
  fi
  iptables -A INPUT -m state --state ESTABLISHED -j ACCEPT

# 4. admit ICMP types
  AICMP="0 3 3/4 4 11 12 14 16 18"
  for tyicmp in $AICMP
  do
     iptables -A INPUT -i $EXTIF -p icmp --icmp-type $tyicmp -j ACCEPT
  done

# 5. admit services types
# iptables -A INPUT -p TCP -i $EXTIF --dport  22  -j ACCEPT   # SSH
# iptables -A INPUT -p TCP -i $EXTIF --dport  25  -j ACCEPT   # SMTP
# iptables -A INPUT -p UDP -i $EXTIF --sport  53  -j ACCEPT   # DNS
# iptables -A INPUT -p TCP -i $EXTIF --sport  53  -j ACCEPT   # DNS
# iptables -A INPUT -p TCP -i $EXTIF --dport  80  -j ACCEPT   # WWW
# iptables -A INPUT -p TCP -i $EXTIF --dport 110  -j ACCEPT   # POP3
# iptables -A INPUT -p TCP -i $EXTIF --dport 443  -j ACCEPT   # HTTPS
```

图 3-16　连接 Internet 外网的配置部分(#3～#5)

❸ 如图 3-17 所示，演示了连接内部局域网的配置，分成 4 个部分完成(见图中脚本＃1～＃4 的注释标号)。第 1 部分，载入了一些必需的模组，如 ip_tables、iptable_nat 等；第 2 部分，清除了 NAT 表的所有规则，并指定了 NAT 表包含的三条链(PREROUTING、POSTROUTING 和 OUTPUT)的默认策略；第 3 部分，是配置 NAT 主机，将在 3.5 节“配置 NAT 网关”中详细讲述，这里暂时不作解释；注释部分 Two Line MTU 下面的两行命令是可选命令，如果取消了注释标记符“＃”，就启动了 MTU 限制范围；第 4 部分，也是可选命令，用来设定内部服务器监视进程。

```
# LAN
# 1. load useful modules
  modules="ip_tables iptable_nat ip_nat_ftp ip_nat_irc ip_conntrack
ip_conntrack_ftp ip_conntrack_irc"
  for mod in $modules
  do
        testmod=`lsmod | grep "${mod} "`
        if [ "$testmod" == "" ]; then
                modprobe $mod
        fi
  done

# 2. clear NAT table
  iptables -F -t nat
  iptables -X -t nat
  iptables -Z -t nat
  iptables -t nat -P PREROUTING  ACCEPT
  iptables -t nat -P POSTROUTING ACCEPT
  iptables -t nat -P OUTPUT      ACCEPT

# 3. open route
  if [ "$INIF" != "" ]; then
    iptables -A INPUT -i $INIF -j ACCEPT
    echo "1" > /proc/sys/net/ipv4/ip_forward
    if [ "$INNET" != "" ]; then
      for innet in $INNET
      do
        iptables -t nat -A POSTROUTING -s $innet -o $EXTIF -j MASQUERADE
      done
    fi
  fi
  # Two Line MTU
  # iptables -A FORWARD -p tcp -m tcp --tcp-flags SYN,RST SYN -m tcpmss \
  #          --mss 1400:1536 -j TCPMSS --clamp-mss-to-pmtu

# 4.
# iptables -t nat -A PREROUTING -p tcp -i $EXTIF --dport 80  \
#          -j DNAT --to 192.168.1.210:80
```

图 3-17　连接内部局域网的配置部分

❹ 以上三步就完成了 Linux 防火墙主要脚本的书写，大部分语句都具有通用性，应用场景变换后无需更改。值得注意的是，第一步基本网络配置部分中的三个参数(EXTIF、INIF 和 INNET)要根据实际网络状况来设定，且务必正确。

❺ 另外，还需要设定一些允许进入或必须防御的 IP 地址或网域，这要通过另外两个脚本 deny 和 allow 来完成。如图 3-18 所示，显示了 deny 脚本内容，仿照图中所示语句的命令格式可以在下面添加需要防御的不信任主机地址或网域。类似地，如图 3-19 所示，显示了 allow 脚本内容，在图中所示命令格式下面也可以任意添加信任的允许进入的 IP 地址或网域。

```
[root@seugrid3 iptables]# vi deny
#!/bin/bash
 iptables -A INPUT -i $EXTIF -s 172.18.21.0/24 -j ACCEPT
~
~
```

图 3-18　编辑 deny 脚本

```
[root@seugrid3 iptables]# vi allow
#!/bin/bash
 iptables -A INPUT -i $EXTIF -s 172.18.21.138 -j DROP
~
~
```

图 3-19　编辑 allow 脚本

❻ 在所有的 shell 脚本都配置好之后，可以使用 sh 命令来执行脚本文件，但需要注意的是，在执行命令前需要将文件权限改为可执行，这使用 chmod 命令来完成。如图 3-20 所示，演示了 shell 脚本文件修改文件权限以及运行脚本文件的命令格式，两种命令格式具体分列于下：

```
chmod 700 [脚本文件名]
sh [脚本文件名]
```

```
[root@seugrid3 iptables]# chmod 700 allow
[root@seugrid3 iptables]# chmod 700 deny
[root@seugrid3 iptables]# chmod 700 rules
[root@seugrid3 iptables]# sh rules
```

图 3-20　执行脚本

❼ 最后总结所设计的防火墙的处理流程，如图 3-21 所示。首先，根据网络实际情况设置网络基本参数，即 EXTIF、INIF 和 INNET，然后再定义核心网络功能。接着就是设定 iptables 的一系列默认策略及规则，启动防御模组，设置允许进入的 ICMP 类型和服务类型。所有配置工作做好后，流入的数据包按照一定顺序与这些规则、策略进行比较，如果完全匹配则允许进入内部局域网；如果没有一条规则能够完全匹配则丢弃。

📖**点评与拓展**：在学习完 iptables 基本的策略配置、规则配置后，又架设了一个简易的 Linux 防火墙，并且配置脚本具有普适性，稍做修改就能适用于其他拓扑结构的网络。

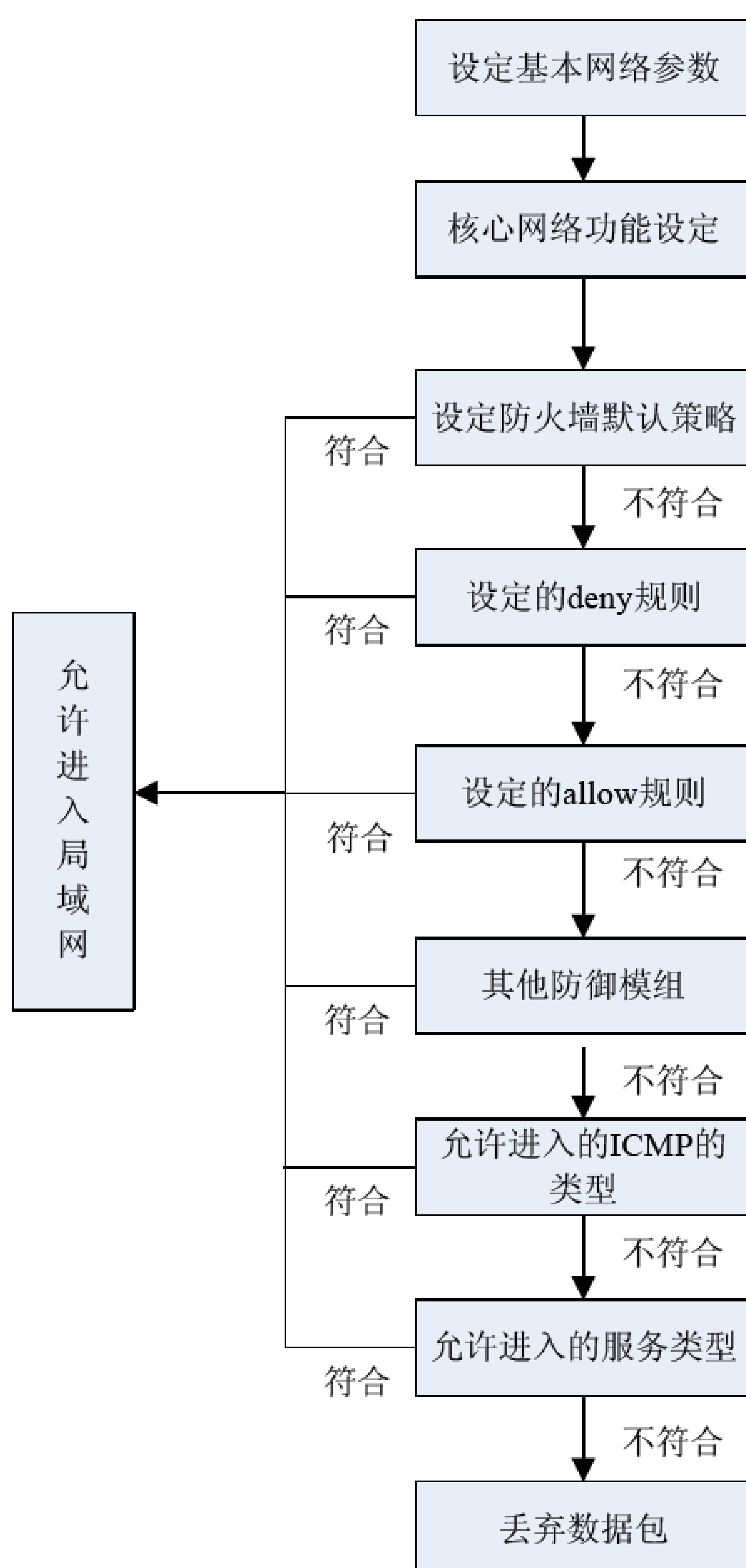


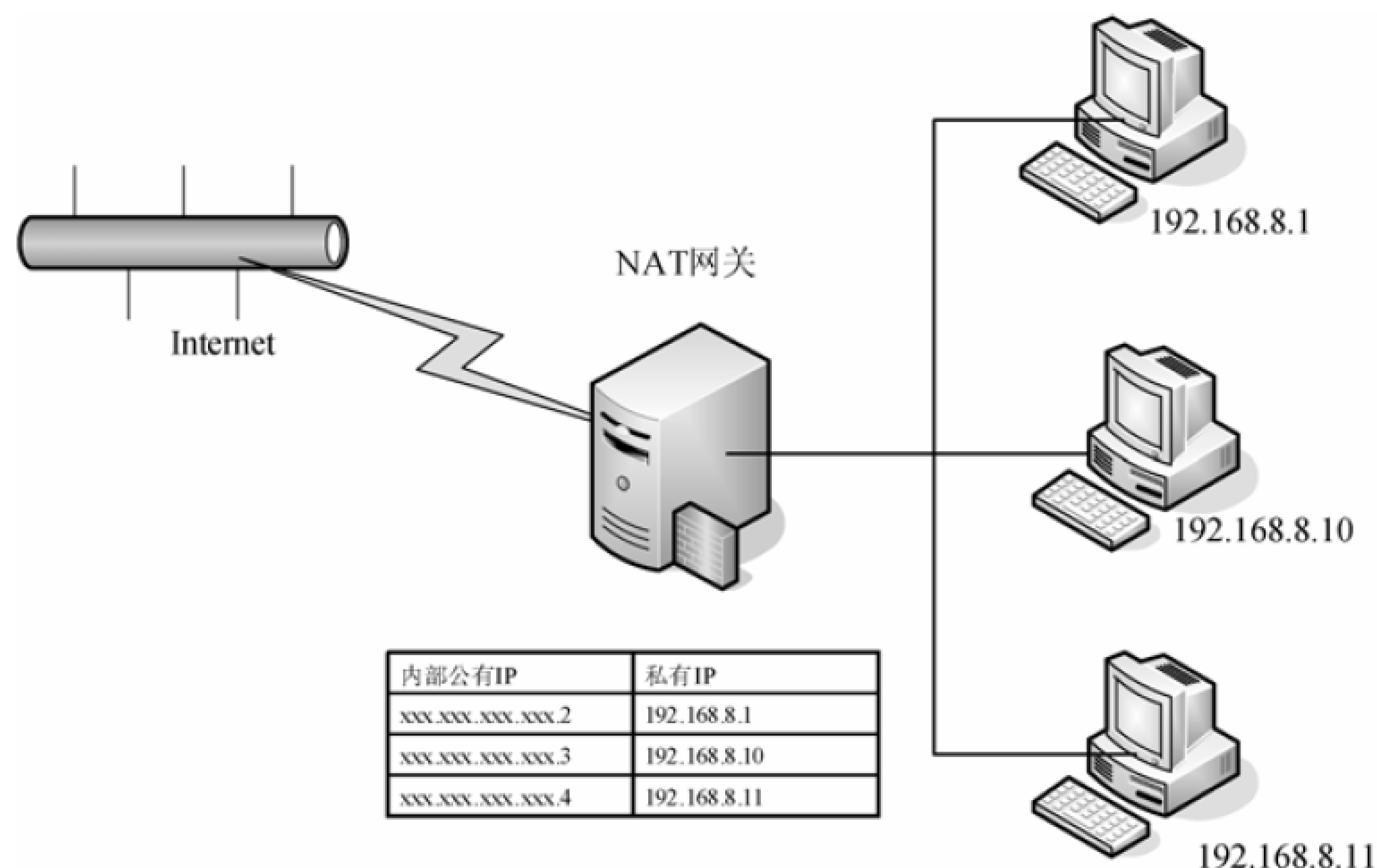
图 3-21　防火墙处理流程

3.4　NAT 服务概述

目前，32 位的 IP 地址资源正逐渐耗尽，因此许多组织机构都采用 1 个外部公有 IP 地址带动一个局域网的方法，局域网内部使用私有 IP。这时就需要使用 NAT(Network Address Translator，网络地址转换)来实现内部私有 IP 与外部公有 IP 之间的转换。

私有地址(Private Address)属于非注册地址，使用私有地址的主机无法直接连到 Internet，只能连接内部局域网(第 2 章详细介绍了私有地址的划分以及子网掩码的概念)，本章将讨论重点放在私有地址如何转化成外部地址去访问 Internet。NAT 就是将一个地址域映射到另一个地址域的标准方法，它能够将内部局域网所有主机的 IP 地址转换成一个外部 IP 地址，从而允许私有地址主机以一个外部公有 IP 地址访问 Internet。

NAT 服务通常部署在网关上，网关是一台装配有两块网卡的服务器，一块网卡使用外部 IP 地址连接到 Internet，另一块网卡使用内部 IP 地址连接内部局域网。当内外部网络之间进行数据传送时，网关就通过查询 NAT 映射表获得相对应的 IP 地址。如果这个 NAT 映射表是网络管理员手工建立的，就称为静态网络地址转换；如果这个 NAT 映射表是网关自动建立的，对网络管理员和用户是透明的，则称为动态网络地址转换。两种 NAT 地址转换方式的基本原理如图 3-22 所示。



内部公有IP	私有IP
xxx.xxx.xxx.xxx.2	192.168.8.1
xxx.xxx.xxx.xxx.3	192.168.8.10
xxx.xxx.xxx.xxx.4	192.168.8.11

图 3-22　NAT 地址转换原理图

这里以动态网络地址转换为例介绍 NAT 服务的工作流程，静态网络地址转换与其不同之处在于 NAT 映射表是由网络管理员预先建好的。具体工作流程如下。

❶ 当内部主机(地址为 192.168.8.10)需要建立一条到 Internet 的会话连接时，首先将请求发送至 NAT 网关上，NAT 网关收到这条请求后，根据接收到的请求数据包检查 NAT 映射表。

❷ 如果还没有为该内部主机建立地址转换映射项，NAT 网关会决定对该地址进行转换，建立 xxx.xxx.xxx.2 ⟷ 192.168.8.10 这个映射项，记录会话状态；如果已经存在该映射项，则 NAT 服务器使用该映射项进行地址转换，并记录会话状态。然后 NAT 网关使用转换后的地址发送数据包至 Internet 公网上。

❸ Internet 公网上的某台主机收到信息后，进行应答，并将应答信息返回给 NAT 网关。

❹ 当 NAT 网关收到应答信息后，先检查 NAT 映射表，如果 NAT 映射表存在匹配项，则使用私有 IP 地址替换 IP 数据包的目的 IP 地址，并将数据包转发给内部主机。如果不存在匹配的映射项，则将数据包丢弃。

第 4 步中，从 Internet 发回的数据包如果没有找到匹配项，就将数据包丢弃，这意味着只要内部主机不主动与 Internet 主机进行通信，Internet 主机将无法访问到内部主机。利用这一原理，NAT 也被用到防火墙技术中，它可以将个别 IP 地址隐藏起来不被外部发现，使外部无法直接访问内部网络设备。

3.5 配置 NAT 网关

应用实例导航——利用 NAT 表配置 NAT 网关

※场景呈现

需要为 A 公司的一个网关配置 NAT 主机，该公司的网络拓扑结构如图 3-13 所示。假设网关的内部 IP 地址为 192.168.1.1，A 公司局域网的内部主机 IP 可以在 192.168.1.2～192.168.1.254 之间的 IP 地址段内任意选择；而网关的外部 IP 地址为 211.65.63.146。现在要求 A 公司局域网内的任一内部主机能够通过 NAT 转换，以网关的外部 IP 地址连接上 Internet。另外，假设该公司的对外主页放在内部 IP 地址为 192.168.1.199 的 Web 服务器上，并且 Web 端口使用的是 8080，要求合理配置 NAT 网关，使得 Internet 上的外部主机能够使用 Internet Explorer 浏览器默认的 80 端口成功访问这台具有内部 IP 的 Web 服务器。

※技术要领

(1) 配置 NAT 表的 POSTROUTING 链。
(2) 配置 NAT 表的 PREROUTING 链。
(3) 配置局域网内的主机。

3.5.1 NAT 网关的基本配置

NAT 主机起到外部 IP 地址和内部 IP 地址的转换功能，NAT 服务一般运行在局域网的网关上。了解了 NAT 服务的功能后，本节主要论述 NAT 主机是如何配置的，分为两块内容：NAT 网关主机的配置和局域网内 PC 机的配置。基本的步骤如下。

❶ NAT 网关主机的配置通过 iptables 的 NAT 表的 POSTROUTING 链的配置来完成。3.3 节没有详细论述 rules 脚本文件关于 NAT 配置部分的内容，这里将详细地进行论述。如图 3-23 标记部分所示，这三条语句是配置 POSTROUTING 链的关键语句：第一条语句让 NAT 网关允许接受所有来自内部局域网的 IP 数据包，即完全信任内部局域网的主机；第二条语句让网关具有转发 IP 数据包的能力，即具有路由功能；第三条语句是对内部 IP 包进行 NAT 地址转换，其中的参数 MASQUERADE 说明将接收到内部 IP 的数据包封装成 eth1 这块网卡上的对外 IP 发送出去。三条命令的格式分别为：

```
iptables -A INPUT -i [内部网卡号] -j ACCEPT
echo "1">/proc/sys/net/ipv4/ip_forward
iptables -t nat -A POSTROUTING -s [私有 IP] -o [外部网卡号] -j MASQUERADE
```

```
# 2. clear NAT table
  iptables -F -t nat
  iptables -X -t nat
  iptables -Z -t nat
  iptables -t nat -P PREROUTING  ACCEPT
  iptables -t nat -P POSTROUTING ACCEPT
  iptables -t nat -P OUTPUT      ACCEPT

# 3. open route
  if [ "$INIF" != "" ]; then
    iptables -A INPUT -i $INIF -j ACCEPT
    echo "1" > /proc/sys/net/ipv4/ip_forward
    if [ "$INNET" != "" ]; then
      for innet in $INNET
      do
        iptables -t nat -A POSTROUTING -s $innet -o $EXTIF -j MASQUERADE
      done
    fi
  fi
  # Two Line MTU
  # iptables -A FORWARD -p tcp -m tcp --tcp-flags SYN,RST SYN -m tcpmss \
  #          --mss 1400:1536 -j TCPMSS --clamp-mss-to-pmtu
```

图 3-23　配置 POSTROUTING 链

❷ 配置好 POSTROUTING 链后，内部主机就能够顺利访问 Internet 了，但是如何使 Internet 上的主机能够顺利访问局域网内的 Web 服务器呢？这时需要通过配置 NAT 表的另一根链 PREROUTING 来完成，图 3-24 演示了 PREROUTING 链的配置命令。

```
Iptables-t nat -A PREROUTING -p tcp -I eth1 --dport 80 -j DNAT --to 192.168.1.199:80
iptables -t nat -A PREROUTING -p tcp -I eth1 --dport 80 -j REDIRECT --to-ports 8080
```

第一条命令说明所有从网卡号 eth1 上传来的访问 80 端口的 TCP 包，都发送到内部主机 IP 为 192.168.1.199 的 80 端口上，实际上，这里将命令中目标的 80 端口改成 8080 就达到了预定的要求；但是为了说清楚问题，特地使用了重定向命令，即第二条命令，它的意思是，所有从网卡号 eth1 上传来的访问 80 端口的 TCP 包，都重定向到内部局域网的 8080 端口。另外需要注意的是，命令中有的参数前是一条横杠 “-” ，有的参数前是两条横杠 “--” ，读者有必要仔细区分，否则 Linux 系统会报无此命令的错误。

```
[root@seugrid3 ~]# iptables -t nat -A PREROUTING -p tcp -i eth1 --dport 80 -j DNAT --to 192.168.1.199:80
[root@seugrid3 ~]#
[root@seugrid3 ~]#
[root@seugrid3 ~]# iptables -t nat -A PREROUTING -p tcp -i eth1 --dport 80 -j REDIRECT --to-ports 8080
[root@seugrid3 ~]# 
```

图 3-24　配置 PREROUTING 链

❸ 最后只要在局域网的主机上进行简单网络配置就可以完成 NAT 网关的配置了，根据场景要求，将 NETWORK(即局域网 IP 段)设为 192.168.1.0，NETMASK(即子网掩码)设为 255.255.255.0，BROADCAST(即广播地址)设为 192.168.1.255，IP 可以在 192.168.1.2～192.168.1.254 范围内任意选取；GATEWAY(即网关地址)设为 NAT 主机的 IP：192.168.1.2；DNS 需要根据公司的 ISP 来设定。这些基本的设定方法在第 2 章的 Linux 网络基本配置中曾经详细讲述过，在此不再赘述。

📖**点评与拓展：**Web 服务器通常默认配置为 8080 端口，但是，Windows 下面 IE 默认的端口是 80，为了避免用户在输入网址后还要加上端口号的麻烦，服务器就使用 iptables 的-j REDIRECT 功能进行端口转换。

3.5.2 NAT 网关的一组技巧性配置

应用实例导航——NAT 网关的辅助配置

※场景呈现

假设 A 公司在完成 NAT 主机和包过滤防火墙的配置后，还需要控制公司内部员工的上网行为，以提高公司员工的工作效率。具体限制要求包括，禁止内部员工访问不健康网站、禁止内部员工开启 QQ 工具聊天、强制内部员工访问被指定的公司主页。要求网络管理员通过添加 NAT 网关上的防火墙的规则达到以上三个要求。

※技术要领

(1) 禁止内部主机访问指定 IP。
(2) 强制内部主机访问指定 IP。

本节一步一步地介绍一组技巧性配置来实现 A 公司的这三个特定需求，希望读者能够在这三个实例的基础上举一反三，通过组合这些简单规则实现更复杂的对内部局域网的控制行为。

首先讲述如何禁止内部主机访问不健康网站。通过禁止域名或 IP 地址的办法来实现控制。如果是禁止域名，iptables 会先查询 DNS 服务器中该域名对应的所有 IP 地址，再将它们加入到禁止规则中，因此，采用禁止域名的方式速度会稍微慢些。图 3-25 演示了两条命令的执行情况，即将所有访问禁止网站的数据包都丢弃。

```
[root@seugrid3 ~]# iptables -I FORWARD -d www.playboy.com -j DROP
[root@seugrid3 ~]# iptables -I FORWARD -d 202.17.61.4 -j DROP
[root@seugrid3 ~]#
```

图 3-25　用命令禁止具体的域名或 IP

其次，讲述如何封锁 QQ。为了防止公司员工使用 QQ 聊天，耽误工作时间，封锁 QQ 是 NAT 网关经常要面对的问题。打开 Windows 系统下 QQ 安装目录\QQ 号\目录下的 Config.db 文件，如图 3-26 所示，这个文件大部分是乱码，但是 QQ 服务器地址可以显示，如图中标记部分。只要将这些服务器全部封禁，就可以达到封锁 QQ 的目的。图 3-27 演示了封禁其中三个服务器的命令，读者可以依此类推出封禁其他服务器的命令。

某些公司机构有时需要强制员工访问指定的网页，比如让员工一打开 IE 浏览器就显示公司主页等。这时只要将内部局域网所有访问 Web 服务器的请求都重定向到指定主页即可，需要注意的是命令中只能使用网页的 IP 地址。图 3-28 显示了该命令的执行。

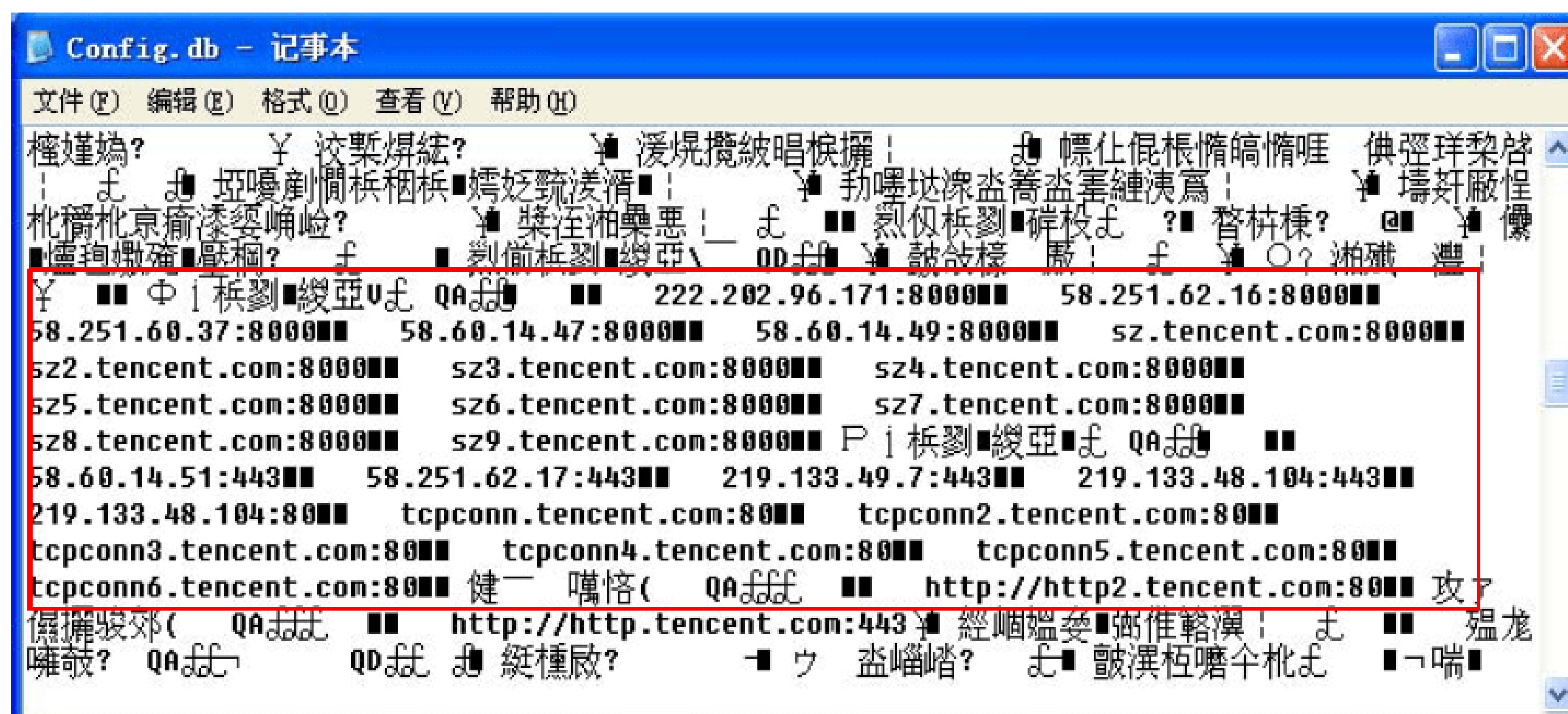


图 3-26　QQ 的 Config.db 文件

```
[root@seugrid3 ~]# iptables -I FORWARD -d tcpconn.tecent.com -j DROP
[root@seugrid3 ~]# iptables -I FORWARD -d tcpconn2.tecent.com -j DROP
[root@seugrid3 ~]# iptables -I FORWARD -d tcpconn3.tecent.com -j DROP
[root@seugrid3 ~]#
```

图 3-27　封锁 QQ 服务器

```
[root@seugrid3 ~]# iptables -t nat -I PREROUTING -i eth0 -p tcp --dport 80 -j DNAT --to 211.65.63.146:9080
[root@seugrid3 ~]#
```

图 3-28　执行脚本(重定向到指定的主页)

点评与拓展：注意，本节的规则都使用了-I 命令，因为-I 命令是在第 1 条规则前插入，没有覆盖后面加入的规则，而这些控制客户端上网的规则都是同时需要满足的，因此不能使用-A 命令。

3.6　本 章 小 结

本章介绍了 Linux 服务器的包过滤防火墙，它利用 iptables 内设的 3 张表来实现包过滤功能。在对 iptables 中的规则、链和表等概念进行了解的基础上，首先介绍了 iptables 基于链的数据包处理流程，揭示了防火墙实现的本质。接着，介绍了如何添加 iptables 链中的规则、如何添加 TCP/UDP 及 ICMP 相关协议的规则；然后，通过一个架设防火墙的实例讲述了如何进行防火墙网络配置、内部局域网配置和外部网络配置的问题，并给出了一个通用的脚本，稍做修改就能适用于其他拓扑的网络环境。

在熟悉了 iptables 的基础上，又介绍了如何利用 iptables 来配置 NAT 网关及一些相关的技巧。配置 NAT 网关是目前很多机构必不可少的技术，具有很强的实用性。

第 4 章　远程控制服务：Telnet、SSH 和 VNC

一台连上 Internet 的服务器，首先要解决的问题就是如何让网络管理员远程登录该服务器，并且进行远程控制。所谓远程控制，是指由一台计算机去控制另一台计算机，并可以管理远程服务器上的软、硬件资源，就如同网络管理员坐在服务器前操作一样的技术。目前，大部分公司和机构或者将服务器放置于 ISP 托管的机房内，或者将本公司和机构的服务器统一放置在一间机房内。这些现实情况都要求借助远程控制的方式对这些地理位置上非常分散的服务器进行统一管理。

在早期的 UNIX/Linux 服务器上，几乎都提供 Telnet 这个远程联机服务器软件。不过，Telnet 是以明码形式来传输数据的，安全性得不到很好的保障。于是，基于著名加密算法 RSA 的网络传输协议族 SSH 应运而生，它很好地保证了数据在传输过程中不被轻易破坏、泄露和篡改，并且有效地防止了桥接攻击。但是，SSH 纯文字接口登录主机进行操控的方式并不能满足操控图形化软件的需求，因此还需要有 VNC 这样的服务来实现图形接口的登录。

通过本章的学习，读者应掌握以下主要内容：

- ✧ Telnet 服务的安装和启动
- ✧ RSA 公钥加密体系架构的原理
- ✧ SSH Server 的启动和配置
- ✧ Windows 平台上 SSH 方式的登录和操控
- ✧ VNC Server 的启动和配置
- ✧ Windows 平台上通过 VNC Viewer 的图形接口登录

4.1　Telnet 服务

4.1.1　Telnet 概述

经常使用 CTerm 软件登录 BBS 的读者肯定会很熟悉 Telnet，CTerm 软件就是利用 Telnet 协议进行远程登录的，Telnet 是目前 Internet 上应用最广泛的协议之一，使得用户可以方便地使用远程主机上的软硬件资源。Telnet 也是最早被使用的远程控制协议，它为早期的网络管理提供了便利性，几乎各种操作系统都内置了 Telnet 协议的客户端软件，可以说，Telnet 是远程控制方面的先驱协议。

尽管由于 Telnet 在安全性上暴露出越来越多的问题而逐渐退出历史的舞台，但是我们仍然需要了解这个先驱协议的安装、配置和使用。

4.1.2　Telnet 服务器端的安装和配置

Telnet 分为服务器端软件和客户端软件，本节讲述 Telnet 服务器端软件的安装、配置和启动。

❶ Fedora 6 系统默认安装了 Telnet 的客户端软件，但是没有安装 Telnet 服务器端软件，因此需要配置 Telnet 服务器。需要首先安装 Telnet 服务器端软件，将 Fedora 安装包上的 Telnet 服务器端软件的 rpm 包复制到服务器上，rpm 名称为 telnet-server-0.17-37.i386.rpm，直接使用下面命令安装，如图 4-1 所示。

```
rpm -ivh  telnet-server-0.17-37.i386.rpm
```

```
[root@seugrid2 local]# rpm -ivh telnet-server-0.17-37.i386.rpm
warning: telnet-server-0.17-37.i386.rpm: Header V3 DSA signature: NOKEY, key ID 4f2a6fd2
Preparing...                ########################################### [100%]
        package telnet-server-0.17-37 is already installed
You have new mail in /var/spool/mail/root
[root@seugrid2 local]#
```

图 4-1　安装 Telnet 服务

提示安装成功后，可以使用下面命令查看 Telnet 软件的安装情况，如图 4-2 所示。

```
rpm -qa | grep telnet
```

可以看到，该服务器上已经安装了 Telnet 服务器端和客户端的软件。

```
[root@seugrid2 local]# rpm -qa | grep telnet
telnet-0.17-37
telnet-server-0.17-37
[root@seugrid2 local]#
```

图 4-2　查看 Telnet 服务的安装情况

❷ Telnet 服务并不启动独立的守护进程，而是交由 xinetd 管理，xinetd（extended internet services daemon）是系统的超级守护进程，它提供了访问控制、重定向、日志记录、优先级和连接数等安全和管理机制，因此 xinetd 托管了 Telnet 的所有用户请求。

由于 Telnet 交由 xinetd 管理，它的配置文件也放到了/etc/xinetd.d 目录下，名字为 telnet，这个配置文件规定了是否开放 Telnet 服务、开放的权限以及最大连接数等属性，它通过在 xinetd 中定义 Telnet 服务结构体来规定。打开/etc/xinetd.d/telnet 文件，如图 4-3 所示，需要对该文件进行两处修改。第一处，将默认的“distable=yes”改为“distable=no”，表示开放 Telnet 服务；第二处，添加“instances=5”，表示同时允许 5 个客户端连接该 Telnet 服务器，如图 4-4 标注部分所示。

```
[root@seugrid2 ~]# vi /etc/xinetd.d/telnet
```

图 4-3　打开/etc/xinetd.d/telnet 文件

```
# default: on
# description: The telnet server serves telnet sessions; it uses \
#       unencrypted username/password pairs for authentication.
service telnet
{
        flags           = REUSE
        socket_type     = stream
        wait            = no
        user            = root
        server          = /usr/sbin/in.telnetd
        log_on_failure  += USERID
        disable         = no
        instances       = 5
}
```

图 4-4　/etc/xinetd.d/telnet 文件的配置

❸ 由于 Telnet 交由 xinetd 来管理，因此只要使用下面命令重启 xinetd 服务来启动 Telnet 守护进程，如图 4-5 所示。

```
/ect/init.d/xinetd restart
```

```
[root@seugrid2 ~]# /etc/init.d/xinetd restart
Stopping xinetd:                                           [  OK  ]
Starting xinetd:                                           [  OK  ]
[root@seugrid2 ~]#
```

图 4-5　重启 xinetd 服务

使用下面命令可以观察 Telnet 服务是否正常启动，如图 4-6 所示。

```
netstat -tulp | grep telnet
```

```
[root@seugrid2 ~]# netstat -tulp | grep telnet
tcp        0      0 *:telnet                *:*                     LISTEN      1781/xinetd
[root@seugrid2 ~]#
```

图 4-6　观察 Telnet 服务

4.1.3　Telnet 客户端的连接

Telnet 的服务器端配置完毕后，本节分别讲述 Linux 和 Windows 两种系统下 Telnet 客户端的连接。

❶ 一般不使用 root 用户连接 Telnet 服务器，为此可以新建 test 用户以登录 Telnet 服务器。如果是在本机登录，输入下面命令后，shell 提示输入登录用户及密码，验证通过后，就以 test 用户登录了本地服务器，进入 test 用户的根目录/home/test，如图 4-7 所示。

```
telnet localhost
```

❷ 在远程主机可以通过 Telnet 协议登录服务器，下面介绍 Windows 系统下的登录过程。选择【开始】→【运行】命令，弹出如图 4-8 所示的【运行】对话框，输入 cmd 进入 DOS 命令行。输入下面命令尝试登录远程服务器，如图 4-9 所示。

```
telnet 172.18.12.178
```

```
[root@seugrid2 ~]# telnet localhost
Trying 127.0.0.1...
Connected to localhost.localdomain (127.0.0.1).
Escape character is '^]'.
Fedora Core release 6 (Zod)
Kernel 2.6.18-1.2798.fc6xen on an i686
login: test
Password:
[test@seugrid2 ~]$ ls
[test@seugrid2 ~]$
```

图 4-7　本地登录 Telnet 服务器

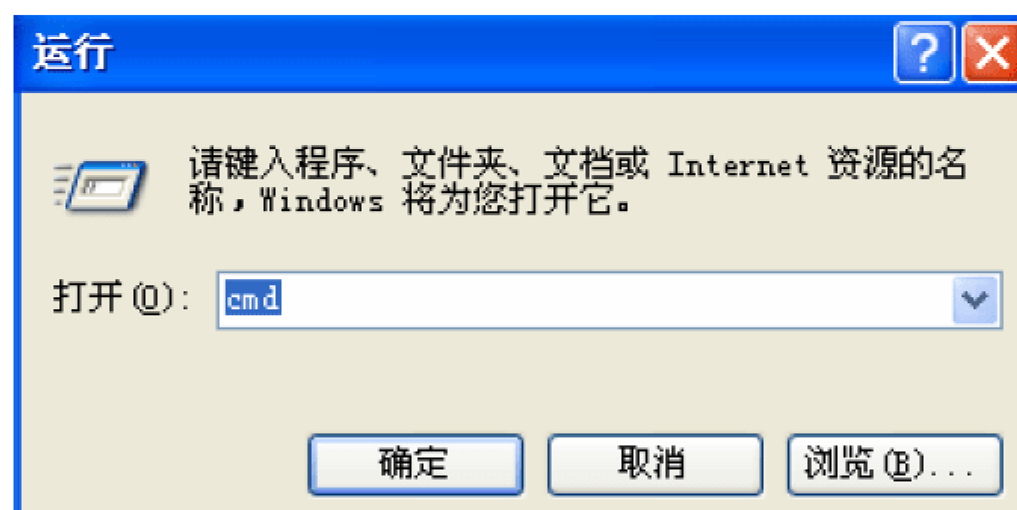


图 4-8　【运行】对话框

```
Telnet 172.18.12.178
Microsoft Windows XP [版本 5.1.2600]
(C) 版权所有 1985-2001 Microsoft Corp.

C:\Documents and Settings\Administrator>telnet 172.18.12.178
正在连接到172.18.12.178...
```

图 4-9　登录 Telnet 服务器

连接上服务器后，提示输入登录用户及密码，验证通过后，就以 test 用户登录了远程服务器，进入 test 用户的根目录/home/test，如图 4-10 所示。

点评与拓展： 由于 Telnet 交由 xinetd 服务管理，因此，安装 Telnet 前需要确保 xinetd 已经安装，而且 Telnet 的启动方式与其他系统服务略有不同。

```
Telnet 172.18.12.178
Fedora Core release 6 (Zod)
Kernel 2.6.18-1.2798.fc6xen on an i686
login: test
Password:
Last login: Thu Nov  8 13:32:07 from 172.18.12.225
[test@seugrid2 ~]$ ls
new
[test@seugrid2 ~]$ _
```

图 4-10　进入 Test 用户的根目录

4.2　SSH 服务

4.2.1　SSH 的概述和原理

传统的网络服务程序，如 ftp、rsh、Telnet 及 rcp 等，由于它们是用明文传送口令和数据，极易被截取和攻击，所以这些网络服务在本质上是不安全的。而且，这些服务程序的安全验证方式也是有其弱点的，就是很容易受到桥接攻击。所谓桥接攻击，就是“中间人”冒充真正的服务器接收你传给服务器的数据，然后再冒充你把做了手脚的数据传给真正的服务器，服务器和客户端之间的数据传送就会出现很严重的问题。

在传统网络服务程序的安全性得不到保障的情况下，SSH(secure shell protocol)应运而生。作为取代 ftp、rsh、Telnet 及 rcp 等的网络连接协议簇，SSH 可以有效防止 IP 地址欺骗、DNS 欺骗和源路径攻击。SSH 提供给用户身份认证的主要方法就是使用公共密钥加密法。根据所用 SSH 版本的不同，可以采用 RSA 或者 Diffie-Hellman 和数字签名标准来实现。也可以选择使用各种不同的身份认证方法，包括公共密钥法、rhosts/shosts 认证法和密码认证法，这些方法都很简单、安全。另外，SSH 还将对传输的数据进行压缩，大幅度加快了传输的速度。

SSH 所提供的是通过网络进入某个特定账号的安全方法。每个用户都拥有自己的 RSA 密钥。通过严格的主机密钥检查，用户可以核对来自服务器的公共密钥同先前所定义的是否一致。这样就防止了某个用户访问一个他没有相应公共密钥的主机。

由于 SSH 提供了主机身份认证，利用公共密钥而不是 IP 地址，所以它使网络更加安全可靠，并且不容易受到 IP 地址欺骗的攻击。这有助于辨认连接到你系统上的访问者身份，从而防止非法访问者登录到你的系统中。表 4-1 列出了 SSH 所能保护的网络攻击。

表 4-1　SSH 保护的网络攻击方式

网络攻击方式	描　述
包欺骗	包的源地址 IP 不是你的，但被伪装成了你的 IP
IP/主机欺骗	IP 或主机名被攻击者盗用
数据泄露	攻击者获得了所传输的数据的明文
数据侦听	攻击者截取你所传输的数据，并分析其内容

那么，SSH 是如何保证传输数据的安全性及客户端的真实性的呢？为了解析其中的原因，首先需要简单介绍公钥加密体制。该体制依赖于一个加密密钥和一个与之相关的不同的解密密钥，并且仅根据密码算法和加密密钥来确定解密密钥在计算上是不可行的，其主要步骤如下。

每一个用户产生一对密钥，用来加密和解密信息。

每一个用户将其中一个密钥存于公开的寄存器或其他可访问的文件中，该密钥即所谓公钥，另一个密钥是私有的，其他用户不可获得它。

若 Bob 要发消息给 Alice，则 Bob 用 Alice 的公钥对消息加密。

Alice 收到消息后，用其私钥对消息解密。由于只有 Alice 知道其自身的私钥，所以其他接收者都不能解密出消息。公钥加密体系的加密过程如图 4-11 所示。

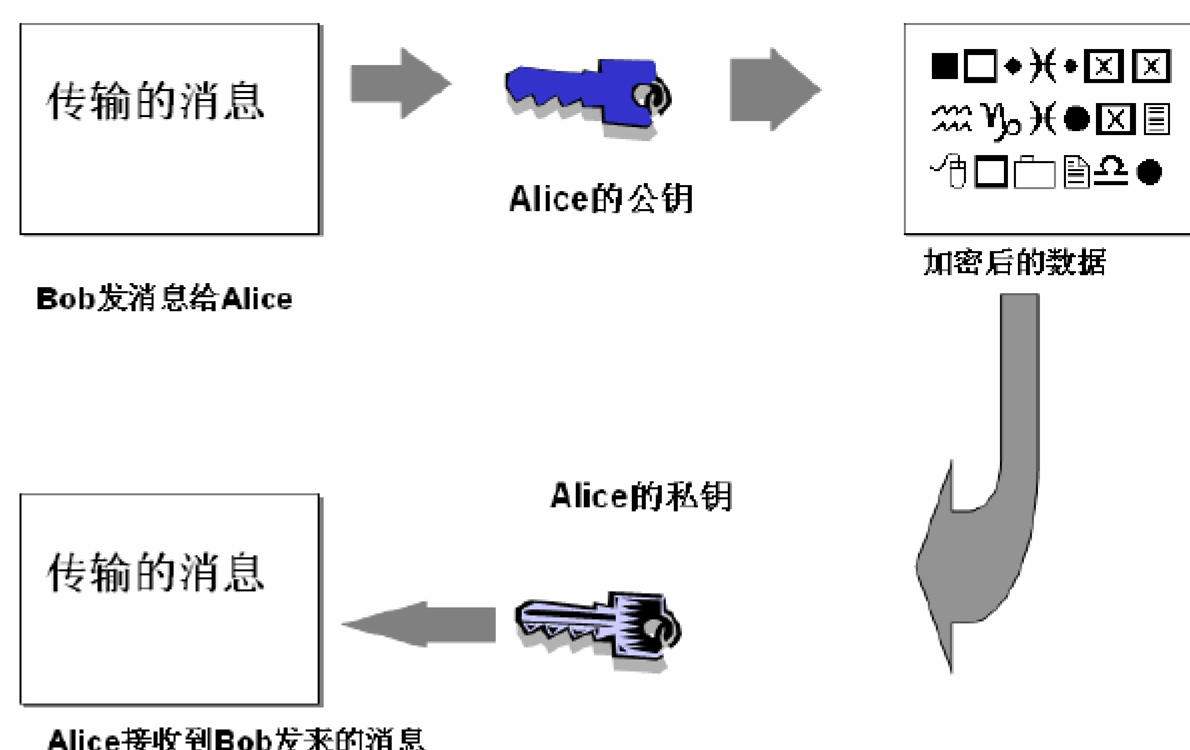


图 4-11　公钥加密体系的加密过程

若 Bob 要发给 Alice 一段认证消息，即让 Alice 确信消息为 Bob 所发，则 Bob 用自身的私钥对消息加密。

Alice 收到消息后，用 Bob 的公钥对消息解密。由于 Alice 可以确定其他人不知道 Bob 的私钥，从而确定消息确为 Bob 所发，如图 4-12 所示。

公钥加密体制具有密钥分配与管理简单、加密传输和数字签名等功能。目前，最流行的是 RSA 非对称算法，它是由 MIT 的 Ron Rivest、Adi Shamir 和 Len Adleman 于 1977 年提出并于 1978 年首次发表。RSA 算法的安全性依赖于大素数难以分解，即将两个大素数相乘十分容易，但是想分解它们的乘积却极端复杂，因此可以将乘积公开作为加密密钥，从一个密钥和密文推断出明文的难度等同于分解两个大素数的积。由于计算机分解大数还没有

有效的实现方法，因此当公钥的长度大于某个长度时，计算机无法在较短有效时间里推导出私钥。RSA 算法自其诞生之日起，就成为被广泛接受且被实现的通用公钥加密方法。

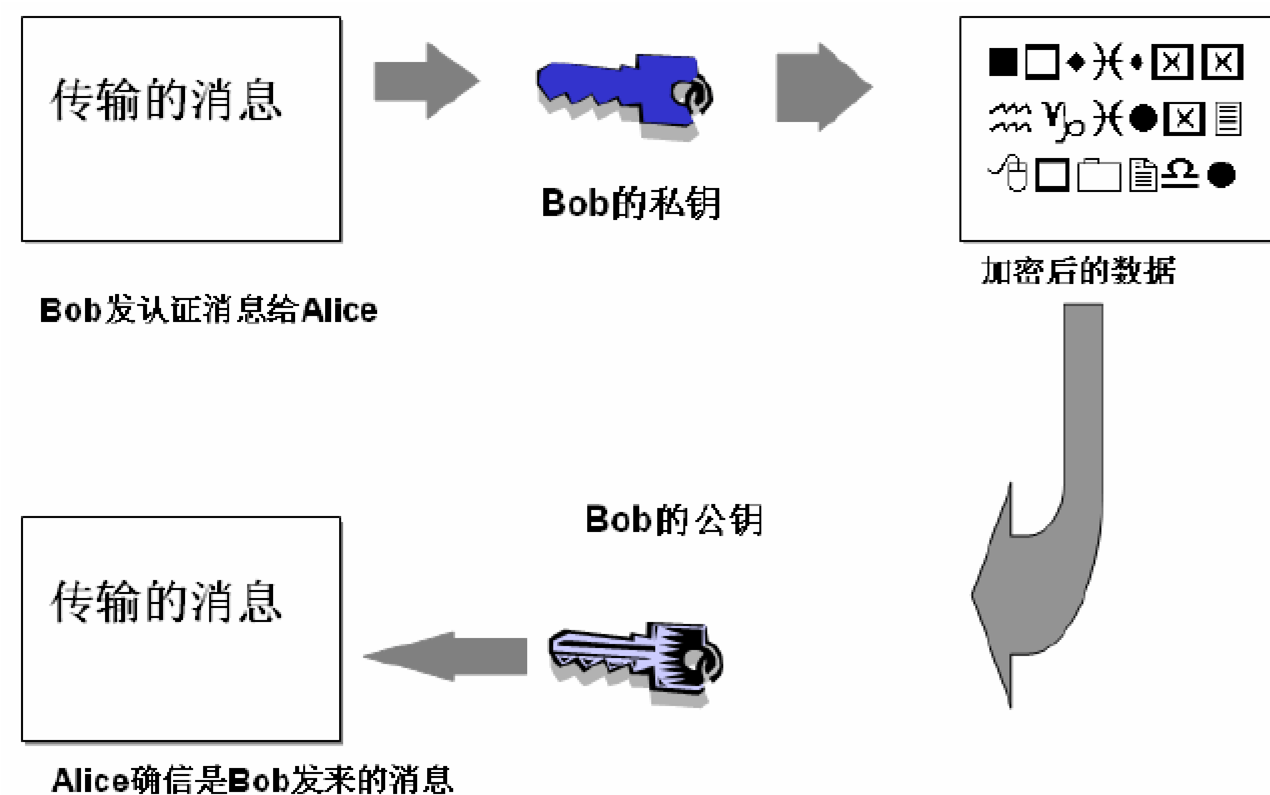


图 4-12　公钥加密体系的认证过程

SSH 包括客户端和服务器程序。服务器程序是一个守护进程，它在后台运行且无需任何类型的常规管理，并响应来自客户端的连接请求。服务器端包含一个文件，即 sshd 程序，它通常被放在目录/ect/init.d 下。服务器端提供了对远程连接的处理，包括公共密钥认证、密钥交换、对称密钥加密和非安全连接本身。对 SSH2 来讲，是用 sftp-server 来管理安全文件传输的连接，其客户端包括几个不同的文件。这些文件包括 ssh、远程复制(scp)和远程登录(slogin)。SSH2 有一个安全文件传输客户端(sftp)，它使用安全文件传输来替代文件传输协议(File Transfer Protocol，FTP)。因为 FTP 不安全，所以 SSH 使用自己的客户端替代它。

接下来对目前流行的 SSH protocol version 2 工作步骤进行描述。

每次 SSH daemon (sshd) 启动时，就会产生一支 768b 的公钥(即 server key)存放在服务器中。

若接收到客户端的 ssh 通信需求时，那么服务器就会将这一支公钥传给 client，此时 client 也会比对一下这支公钥的正确性。比对的方法为利用/etc/ssh/ssh_known_hosts 或~/.ssh/known_hosts 档案内容。

在客户接受这个 768b 的 server key 之后，客户自己也会随机产生一支 256b 的私钥(host key)，并且以加密的方式将 server key 与 host key 整合成一对完整的 Key pair，并且将这对 Key pair 也传送给服务器。

在以后的服务器与客户端的通信中，就以这一对 1024b 的 Key pair 来进行数据的传递；并且在每次通信开始时，服务器端利用 Diffie-Hellman 机制确定客户端的真实性。

以前 SSH protocol version 1 在通信过程中，当服务器端接受客户端的 private key 后，就不再针对该次通信的 Key pair 进行检验。此时若有恶意的 cracker 针对该通信给予恶意的程序代码，由于主机端不会检验通信的正确性，因此可能会接受该程序代码，造成系统被黑等问题。为了弥补这个漏洞，SSH version 2 多加了一个确认通信正确性的 Diffie-Hellman 机制，在每次数据的传输当中服务器端都会以该机制检查资料的来源是否正确，所以可以避免通信过程当中被插入恶意程序代码的问题，进一步加强了 SSH 的安全性。

4.2.2　SSH 服务的启动

在 Fedora 6.0 版本的 Linux 的系统中，已经包含了 SSH 所需要的 OpenSSL 与 OpenSSH 套件，因此就省略了安装 SSH 的过程，而直接可以启动 SSH 的守护进程 sshd。启动 sshd 进程有两种方式：第一种是通过 shell 命令启动 sshd 进程，第二种是通过 Linux 自带的系统配置图形化工具启动 sshd 进程。

首先讲述第一种启动方式，即通过 shell 命令的方式启动，输入下面命令启动或重启动 SSH 服务，如图 4-13 所示。

```
/ect/init.d/sshd start
/ect/init.d/sshd restart
```



图 4-13　启动 SSH 服务

Linux 系统服务的入口程序大都存放在/ect/init.d 目录下，然后通过以下命令格式执行启动、重启动、关闭、显示状态等操作。

```
/etc/init.d/(service)  {start|stop|restart|reload|condrestart|status}
```

接着讲述第二种启动方式，即通过配置图形化工具菜单启动。输入 shell 命令：setup，弹出如图 4-14 所示界面，这就是 Linux 提供的系统配置图形化工具，它方便了初级用户进行显示器配置、网络配置、系统服务和防火墙配置等操作。选择【系统服务】选项，进入图 4-15 所示的界面，该对话框中列出系统提供的所有系统服务，包括 sshd、vsftp、vncserver 等重要服务，只要选择 sshd 选项，打上星号*，单击【确定】按钮就启动了 sshd 进程。

无论使用以上哪种方式启动 SSH 后，都可以通过以下命令查看 SSH 的状态，如图 4-16 所示，ssh daemon 在 23030 号端口处于正常监听状态。

```
netstat -tlp | grep ssh
```

点评与拓展： Linux 系统中的所有服务的启动方式都类似于 SSH 的启动方式，shell 方式的启动命令都类似于 SSH 第一种启动方式中所介绍的。



图 4-14　系统配置图形化工具菜单



图 4-15　启动 SSH 服务

```
[root@seugrid3 ~]# netstat -tlp | grep ssh
tcp        0      0 *:ssh                   *:*                         LISTEN      23030/sshd
[root@seugrid3 ~]#
```

图 4-16　查看 SSH 状态

4.2.3　SSH Secure Shell Client 软件简介

在 Linux 服务器上启动好 SSH daemon 进程后，就可以在 Windows 下方便地对服务器进行远程控制了。我们推荐使用免费软件 SSH Secure Shell Client，该软件提供了快捷方便的 SSH 远程连接和可靠的 FTP 文件传输。

❶ 依照 Windows 傻瓜式 step by step 安装好 SSH Secure Shell Client 后，打开软件，单击 Quick Connect 按钮，弹出如图 4-17 的界面。

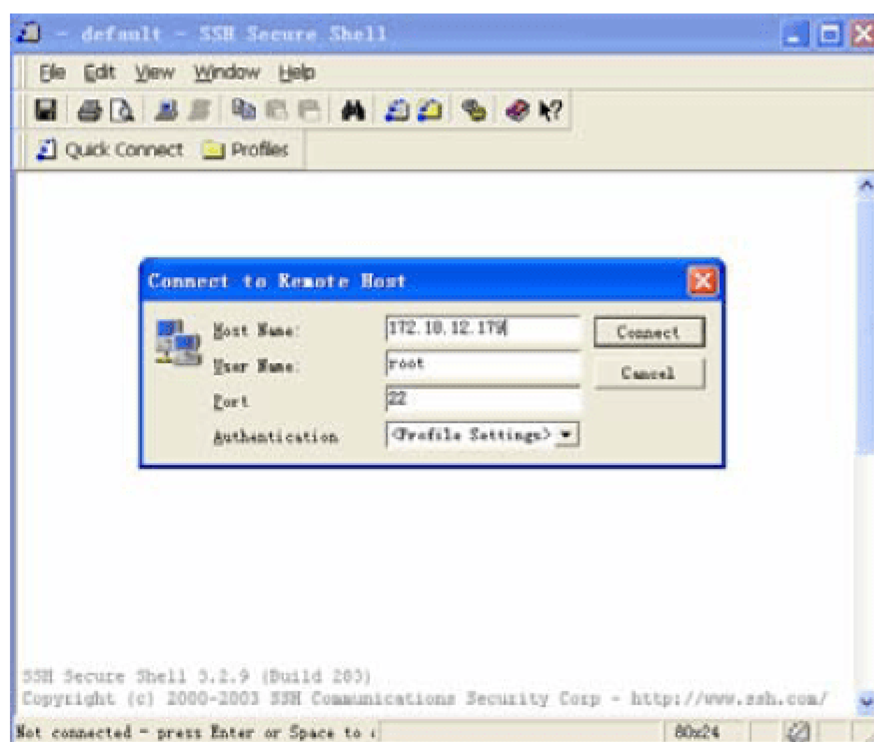


图 4-17　登录界面

❷ 输入需要连接的服务器的 IP 地址以及用户名，Port 和 Authentication 项默认；单击 Connect 按钮，根据提示正确输入密码。如果用户是第一次登录，SSH Secure Shell Client 会弹出询问是否保存公钥文件的窗口，选择确定就进入如图 4-18 所示的 shell 命令提示行。

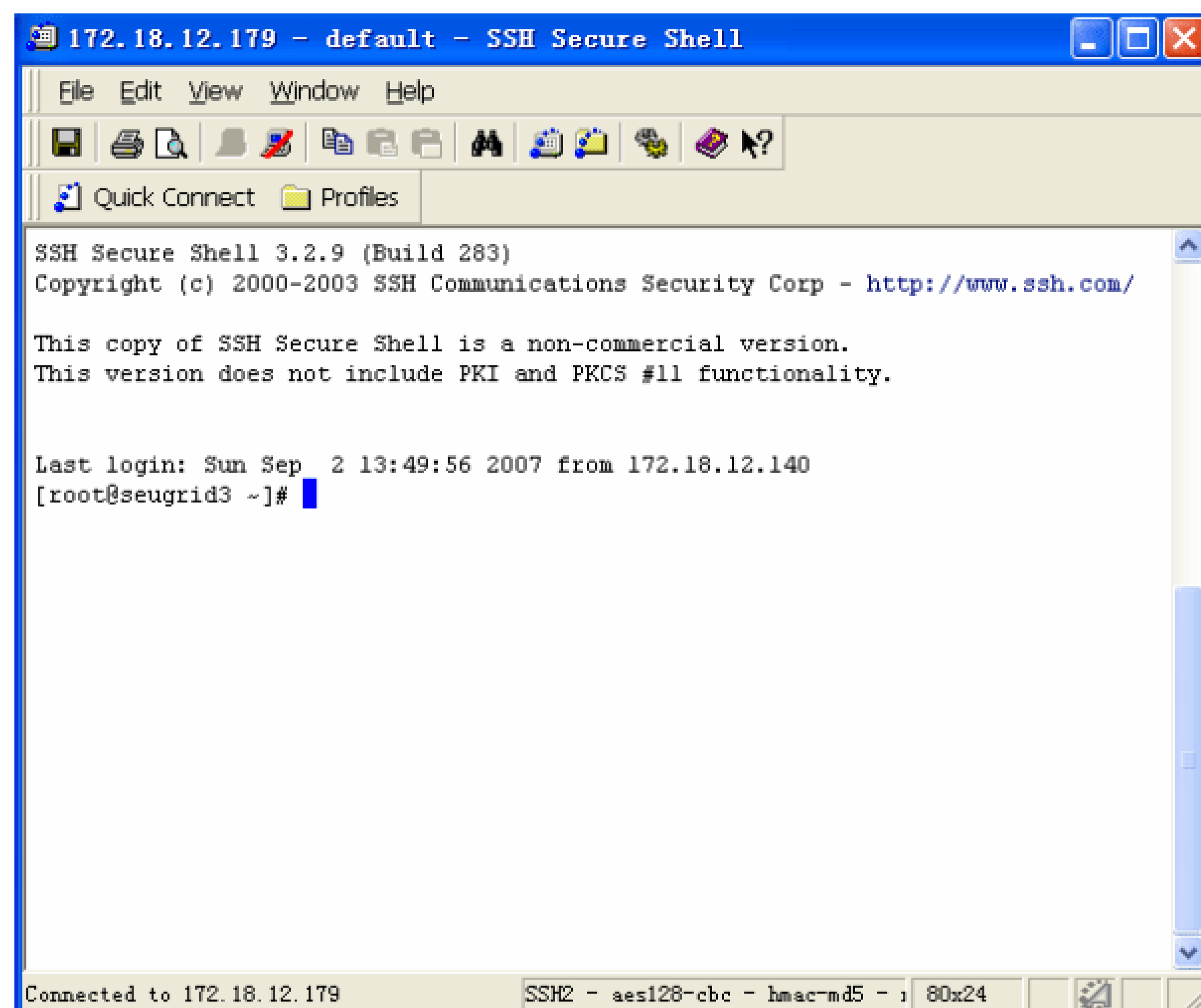


图 4-18　登录成功的命令行

❸ 选择 Edit→Settings 菜单命令进入如图 4-19 所示的软件设置界面，可以根据个人喜好选择不同的背景字体等；图 4-19 中标注部分的 Scrollback buffer 文本框表示了 shell 命令行的最大行数，通常将其设高一些，以避免 shell 命令行的显示信息丢失。

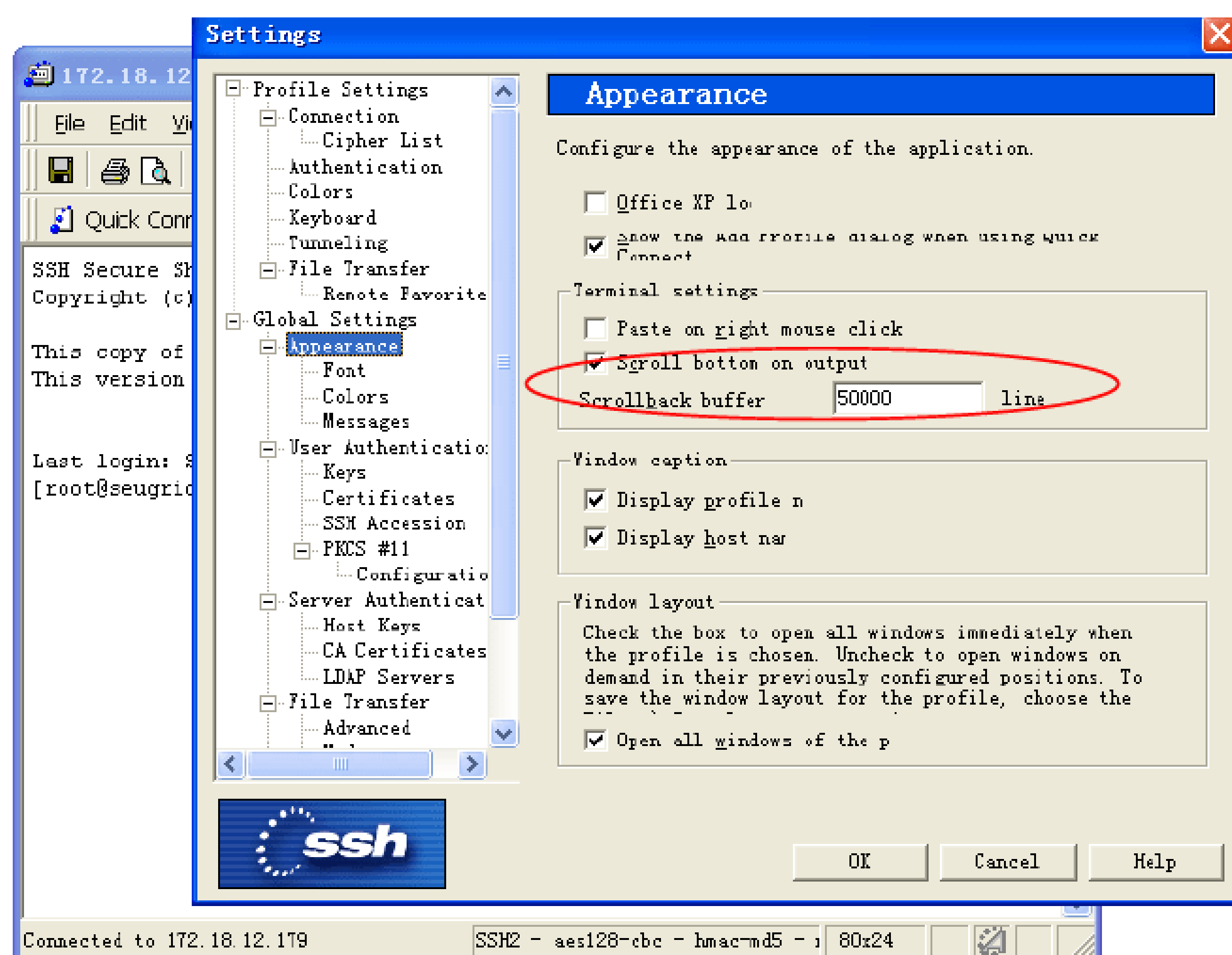


图 4-19　设置界面

❹ SSH Secure Shell Client 还提供了可靠的 FTP 传输工具，方便了 Windows 客户端和 Linux 服

务器端的数据交互，选择 Window→New File Transfer 菜单命令弹出如图 4-20 所示的界面，左半边窗口为 Windows XP 的文件系统，右半边为 Linux 文件系统，从中可以方便地实现双方的数据交互。

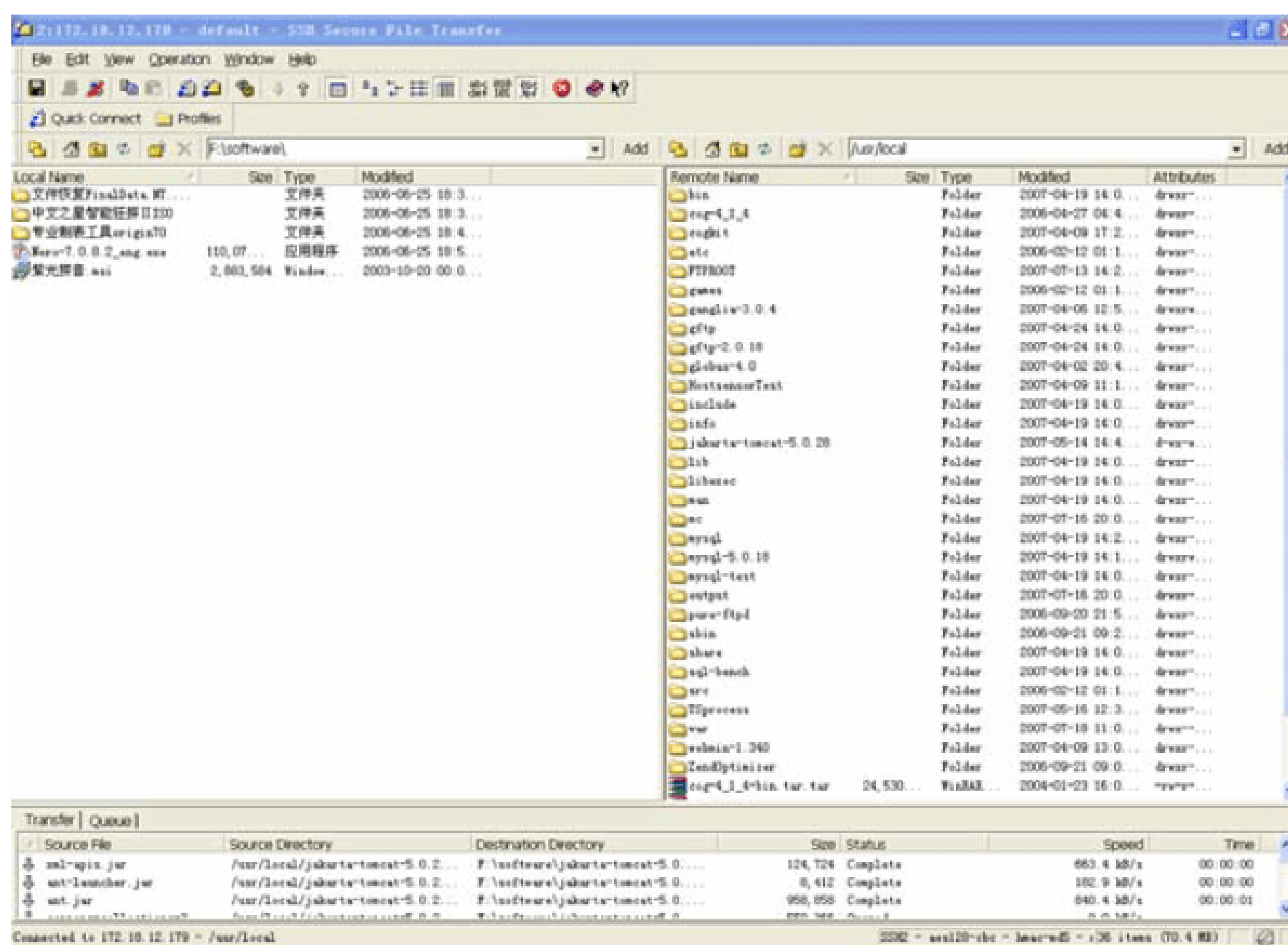


图 4-20　传输文件界面

4.3　配置 SSH 无密码登录

应用实例导航——高性能计算中心的 SSH 无密码登录配置

※场景呈现

假设有某一高性能计算中心，网络拓扑结构如图 4-21 所示，高速局域网 IP 地址段为 172.18.12.0～172.18.12.255，局域网内部署了多台计算结点，其中一台计算结点的 IP 地址为 172.18.12.179；主控结点的以太网 IP 地址为 211.65.63.109，并以此 IP 地址接入 Internet 公网；主控结点的局域网 IP 地址为 172.18.12.178，以此 IP 地址管理其他从属计算结点。现在要求配置 IP 地址为 172.18.12.178 的主控结点，实现以 root 用户无密码 SSH 登录所有从属计算结点，要求以登录 IP 地址为 172.18.12.179 的从属结点为例进行配置。

※技术要领

(1) 详细设定 sshd 配置文件。

(2) 产生主控结点公私钥对，并存放入相应的目录。

(3) 在从属结点根据 sshd 配置文件，将主控结点公钥存放入从属结点的配置文件中。

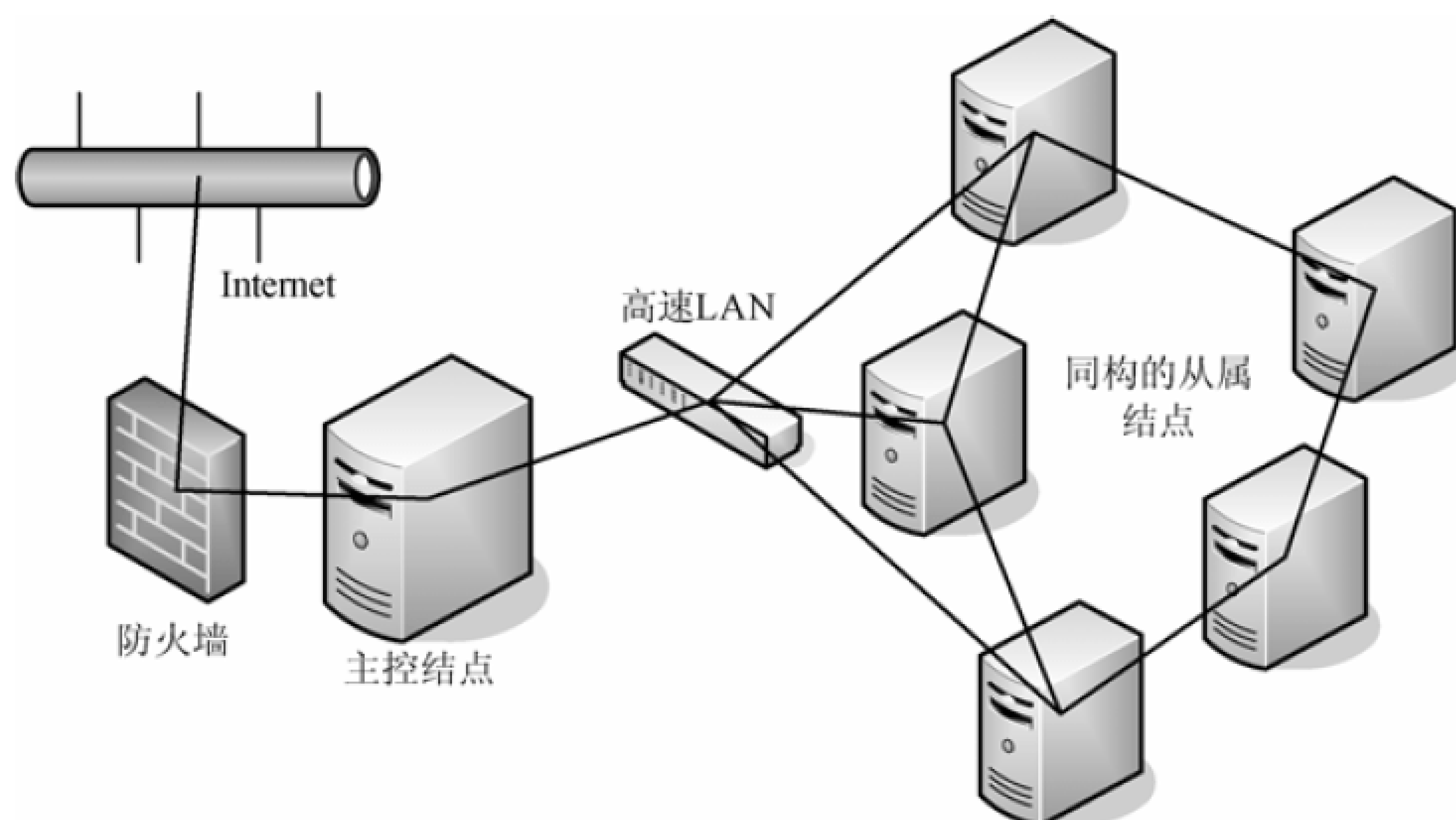


图 4-21　服务器拓扑结构图

SSH 是通过密钥比对数据来验证用户身份的，用户远程登录服务器输入密码时，就自动生成了密钥对。那么人们自然就想到如果预先将客户端所产生的密钥复制到服务器，以后客户端每次登录服务器时，服务器就默认该客户端已经通过了身份验证，而不必重新输入密码进行验证。配置 SSH 无密码登录包含如下 3 个步骤。

为客户端建立自己的公钥和私钥，所使用到的命令为 ssh-keygen;。

将私钥放在客户端的登录用户的根目录的.ssh 目录$HOME/.ssh/下，并且将权限修改为仅有该用户可读；

将公钥放在任何一个想要登录的服务器端的某用户的根目录的.ssh 目录下，即可完成整个配置过程。

下面就按照上述 3 个步骤的思路以 172.18.12.178 和 172.18.12.179 两个结点为例详细讲述 SSH 无密码登录的配置过程。

❶ 由于本例中是实现主控结点无密码登录局域网内的从属结点，因此，主控结点就等同于上述配置步骤中的客户端，从属结点等同于服务器端。首先对 SSH 服务器端进行设置，用 vi 命令编辑从属结点 172.18.12.179 的/etc/ssh/sshd_config 文件，将 AuthorizedKeysFile 行的注释符取消，如图 4-22 所示。AuthorizedKeysFile 规定了无密码登录时，读取客户端公钥的文件的路径。

```
# Authentication:

#LoginGraceTime 2m
#PermitRootLogin yes
#StrictModes yes
#MaxAuthTries 6

#RSAAuthentication yes
#PubkeyAuthentication yes
AuthorizedKeysFile      .ssh/authorized_keys
```

图 4-22　/etc/ssh/sshd_config 文件需添加的语句

❷ 配置好 sshd_config 文件后，就可以在主控结点 172.18.12.178 输入下面命令产生公私钥对，按 Enter 键选择默认设置，如图 4-23 所示，生成的私钥名字为 id_rsa，公钥名字为 id_rsa.pub。

```
ssh-keygen -t rsa
```

```
[root@seugrid2 ~]# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
d5:16:97:43:83:37:47:ff:bb:b3:e8:3d:f6:7c:e3:39 root@seugrid2.seu.edu.cn
```

图 4-23　主控结点产生公私钥对

在/root/.ssh 目录下就能看到刚才所产生的公私钥对，如图 4-24 所示。

```
[root@seugrid2 .ssh]# ll
total 48
-rw-r--r-- 1 root root  395 Jul 12 11:04 authorized_keys
-rw------- 1 root root  668 Jul 11 19:41 id_dsa
-rw-r--r-- 1 root root  614 Jul 11 19:41 id_dsa.pub
-rw------- 1 root root 1675 Sep  4 20:08 id_rsa
-rw-r--r-- 1 root root  406 Sep  4 20:08 id_rsa.pub
-rw-r--r-- 1 root root 1581 Jul 11 16:30 known_hosts
```

图 4-24　公私钥对的存放目录和属性

3 将上一步所产生的公钥复制到从属结点 172.18.12.179 上，复制的方法可以使用下面命令以 rcp 的方式传输；或者使用上一节所介绍的 SSH Secure Shell Client 经过 Windows 系统中转。然后利用 cat 命令将公钥内容附在前面设定的 AuthorizedKeysFile 文件之后。本例中为 authorized_keys 文件，如图 4-25 所示，所用命令分列如下：

```
scp id_rsa.pub root@172.18.12.179 :/root/.ssh
cat id_rsa.pub>>authorized_keys
```

```
[root@seugrid3 ~]# cd /root/.ssh
[root@seugrid3 .ssh]# cat id_rsa.pub>>authorized_keys
[root@seugrid3 .ssh]# ls
authorized_keys  id_dsa  id_dsa.pub  id_rsa  id_rsa.pub  known_hosts
[root@seugrid3 .ssh]#
```

图 4-25　从属结点公钥的放置

4 最后测试在主控结点 172.18.12.178 上登录从属结点，输入下面命令，可以看到无需输入密码便直接登录上了 172.18.12.179 结点，如图 4-26 所示。

```
ssh 172.18.12.179
```

点评与拓展：只要把 172.18.12.178 所产生的公钥文件 id_rsa.pub 逐一复制到局域网内的所有计算从属结点，便可以实现无密码 SSH 登录所有计算结点，这样便省去了每次 SSH 连接都要输入同样密码的重复劳动，大幅度增加了网络管理员的管理效率。

```
[root@seugrid2 ~]# ssh 172.18.12.179
Last login: Tue Sep  4 20:06:46 2007 from seugrid2.seu.edu.cn
[root@seugrid3 ~]# 
```

图 4-26　测试无密码登录从属结点

4.4　VNC 服务的配置和应用

4.4.1　VNC 概述

VNC(virtual network computing)最早是一套由剑桥大学 AT&T 实验室所开发的轻量型可操控远程计算机的软件，其采用了 GPL 授权条款，任何人都可免费取得该软件。VNC 软件主要由两部分组成：VNC Server 及 VNC Viewer。用户需要先将 VNC Server 安装在被控端的计算机上后，才能在主控端执行 VNC Viewer 控制被控端。VNC Server 和 VNC Viewer 支持多种操作系统，包括 Windows、Linux、UNIX、MacOS 等，因此可以利用 VNC 服务实现不同操作系统之间的相互远程控制。

VNC 提供了图形化界面的远程操作，就如同用户亲自坐在被控端前执行被控端的应用程序，并使用被控端的系统资源，这也是它与 SSH 的最大区别。它用于远程启动服务器的图形化应用程序，如 GFtp、Java-COG 等。以往，人们习惯于使用 X-Win32 进行远程操作，X-Win32 底层也采用了 SSH 协议，但是当多用户并行操作时，多个用户共用一个窗口，相互之间存在干扰；VNC 彻底解决了这一不足，为并行操作的用户分别分配了独立窗口，并允许其对 Linux 用户组权限进行配置。整个 VNC 运行的工作流程一般如下。

VNC 客户端通过 VNC Viewer 连接至 VNC Server。

VNC Server 传送一对话窗口至客户端，要求输入联机密码，以及存取的 VNC Server 显示装置。

在客户端输入联机密码后，VNC Server 验证客户端是否具有存取权限。

若是客户端通过 VNC Server 的验证，客户端即要求 VNC Server 显示桌面环境。

被控端将画面显示控制权交由 VNC Server 负责。

VNC Server 将把被控端的桌面环境利用 VNC 通信协议送至客户端，并且允许客户端控制 VNC Server 的桌面环境及输入装置。

4.4.2　VNC 的配置和启动

本节讲述 VNC 服务器端的配置步骤以及 VNC 服务器端监控进程的启动。

❶ Fedora 默认安装了 VNC 服务，可以使用下面命令检查系统是否已经安装了 VNC 服务或查看安装的是何种版本的 VNC 服务。

```
rpm -q vnc-server
```

如图 4-27 所示，VNC 服务已经安装，版本为 4.1.1-36。如果 VNC 服务没有安装，则利用 Fedora 安装盘上的 RPM 包进行安装。

```
[root@seugrid3 ~]# rpm -q vnc-server
vnc-server-4.1.1-36
[root@seugrid3 ~]#
```

图 4-27　检查 VNC 安装包

❷ VNC 服务支持多桌面，它为每个桌面分配一个桌面号，每个用户的连接需要占用一个桌面，每个桌面通过一个 TCP 端口进行通信。VNC 默认 1 号桌面通过 5901 号端口利用 TCP 协议通信，2 号桌面通过 5902 号端口通信，以此类推。如果 Linux 服务器开启了防火墙功能，就需要开放 TCP 协议所对应的端口。因此，在启动 VNC 桌面之前，一定要开启从 5901 开始的 TCP 端口，可以使用下面命令添加防火墙规则（第一条命令开启 5901 一个端口，第二条命令开启从 5901～5910 之间的所有端口），iptables 命令在第 3 章“Linux 防火墙与 NAT 服务”中已经详细介绍过，在此不作解释，仅将命令分列于下:

```
iptables -I INPUT -p tcp  --dport 5901  -j ACCEPT
iptables -I INPUT -p tcp  --dport 5901: 5910  -j ACCEPT
```

VNC 服务所有的配置文件存放在/root/.vnc 目录下，如图 4-28 所示。其中 passwd 文件存放了 VNC 口令的密文；xstartup 是系统自动为用户建立的配置文件，以后每次启动 VNC 服务时，都会读取该文件中的配置选项。

```
[root@seugrid3 ~]# cd .vnc
[root@seugrid3 .vnc]# ls
passwd          sec.seu:2.log  sec.seu:3.pid  seugrid:2.log   seugrid3:5.pid  seugrid:4.log  seugrid:5.pid
sec.seu:1.log   sec.seu:2.pid  seugrid:1.log  seugrid:2.pid   seugrid:3.log   seugrid:4.pid  xstartup
sec.seu:1.pid   sec.seu:3.log  seugrid:1.pid  seugrid3:5.log  seugrid:3.pid   seugrid:5.log
[root@seugrid3 .vnc]#
```

图 4-28　/root/.vnc 目录

❸ 做好准备工作后，我们可以使用命令 vncserver :1 启动 1 号桌面，以此类推。如图 4-29 所示，vncserver :5 启动了尚未启动的 5 号桌面，系统提示成功；当输入 vncserver :1 时，系统提示 1 号桌面已经启动。

```
[root@seugrid3 .vnc]# vncserver :5

New 'seugrid3:5 (root)' desktop is seugrid3:5

Starting applications specified in /root/.vnc/xstartup
Log file is /root/.vnc/seugrid3:5.log

[root@seugrid3 .vnc]# vncserver :1
A VNC server is already running as :1
[root@seugrid3 .vnc]#
```

图 4-29　启动 VNC 桌面命令

❹ 为了方便地管理桌面号与用户之间的对应关系，VNC 服务将这些配置信息写入文件 vncservers 中，该文件存放在/etc/sysconfig 目录下。用 vi 命令打开该文件，添加下面的两条语句，如图 4-30 所示。

```
# The VNCSERVERS variable is a list of display:user pairs.
#
# Uncomment the lines below to start a VNC server on display :2
# as my 'myusername' (adjust this to your own).  You will also
# need to set a VNC password; run 'man vncpasswd' to see how
# to do that.
#
# DO NOT RUN THIS SERVICE if your local area network is
# untrusted!  For a secure way of using VNC, see
# <URL:http://www.uk.research.att.com/archive/vnc/sshvnc.html>.

# Use "-nolisten tcp" to prevent X connections to your VNC server via TCP.

# Use "-nohttpd" to prevent web-based VNC clients connecting.

# Use "-localhost" to prevent remote VNC clients connecting except when
# doing so through a secure tunnel.  See the "-via" option in the
# `man vncviewer' manual page.

# VNCSERVERS="2:myusername"
# VNCSERVERARGS[2]="-geometry 800x600 -nolisten tcp -nohttpd -localhost"
VNCSERVERSERS="1:root 3:root"
VNCSERVERSERS="2:bob"
```

图 4-30　在 vncservers 中添加的内容

```
VNCSERVERSERS="1:root 3:root"
VNCSERVERSERS="2:bob"
```

这两条语句将 1 号和 3 号桌面与 root 用户相绑定，2 号桌面与 bob 用户连接，请注意：bob 用户必须为 Linux 的系统用户。当下次以 root 用户远程登录时，VNC 分配给它 1 号桌面或 3 号桌面，以 bob 用户登录时，分配给它 2 号桌面。VNC 服务每次启动时都会根据这个配置文件自动创建桌面号。

5 利用 vncpasswd 命令可以修改用户登录 VNC 的密码，如图 4-31 所示，root 用户利用 vncpasswd 命令修改了自己的登录密码。如果 bob 用户需要修改密码，则首先在 shell 输入以下命令切换到 bob 用户，然后使用 vncpasswd 修改。

```
su bob
```

```
[root@seugrid3 .vnc]# vncpasswd
Password:
Verify:
[root@seugrid3 .vnc]#
```

图 4-31　更改 VNC 密码

4.4.3　Tight VNC Viewer 软件

应用实例导航——利用 VNC 图形化登录

※场景呈现

某公司网络管理员现在需要为一放置在异地分公司的服务器下载软件，采用远程控制服务器方式，该服务器的 IP 地址为 172.18.12.179，并且要求利用图形化 GFTP 工具登录 IP 地址为 172.18.12.238 的 FTP 服务器。

※技术要领

(1) 利用 Tight VNC Viewer 软件远程操控服务器。
(2) 利用 GFTP 软件登录 FTP 服务器。

本节介绍如何在 Windows 平台上利用 VNC 远程操控服务器，推荐使用 Tight VNC Viewer 软件，该软件是免费软件，下面详细介绍其下载、安装和使用过程。

❶ 用户可以到官方网站 http://www.tightvnc.com/download.html 下载，如图 4-32 所示，官方网站提供了 Tight VNC Viewer 软件的 Windows 和 Linux(Fedora 6.0)版本。

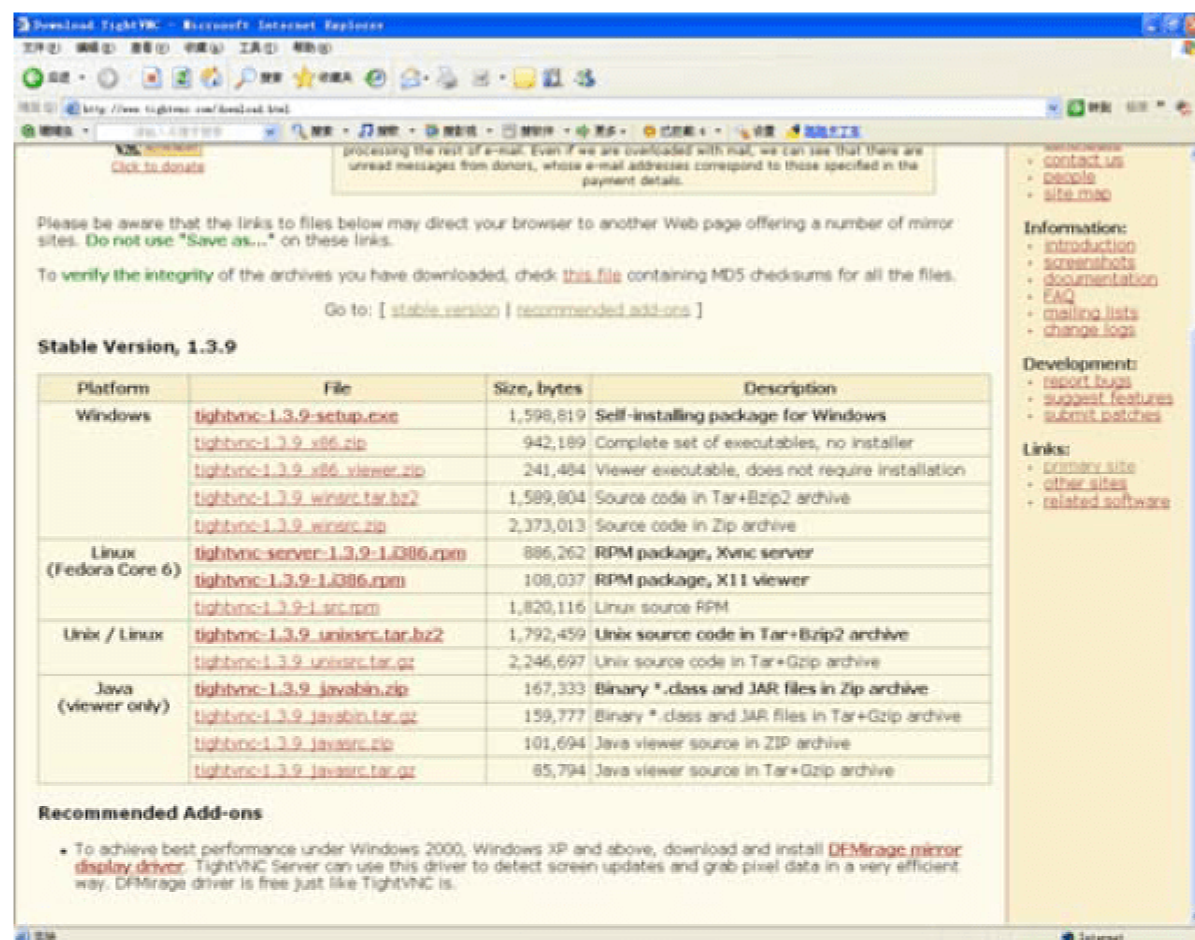


图 4-32　VNC Viewer 下载

❷ 下载完毕之后，安装过程很简单，按照默认值安装即可。需要注意的是，在步骤 Select Components 中，选择 Custom installation 选项，取消选中 TightVNC Server 复选框，即只安装客户端程序，如图 4-33 所示。

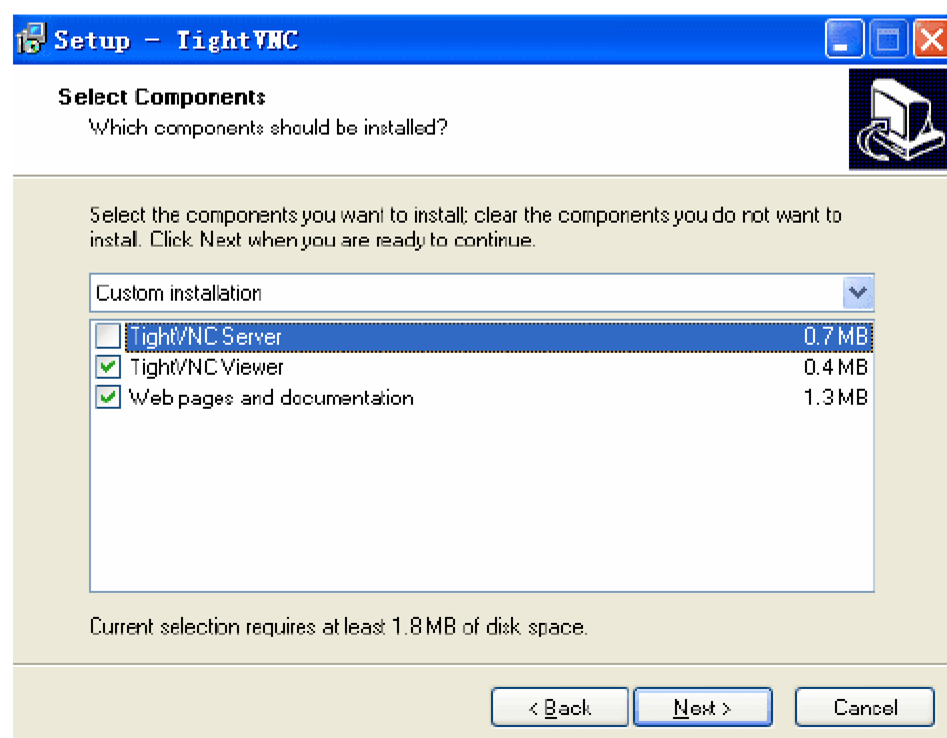


图 4-33　VNC Viewer 安装

❸ 安装完毕之后，打开 Tight VNC Viewer 程序，出现如图 4-34 所示界面，需要用户填写服务器 IP 地址及桌面号，图中所示连接 IP 为 172.18.12.179 服务器的 1 号桌面。

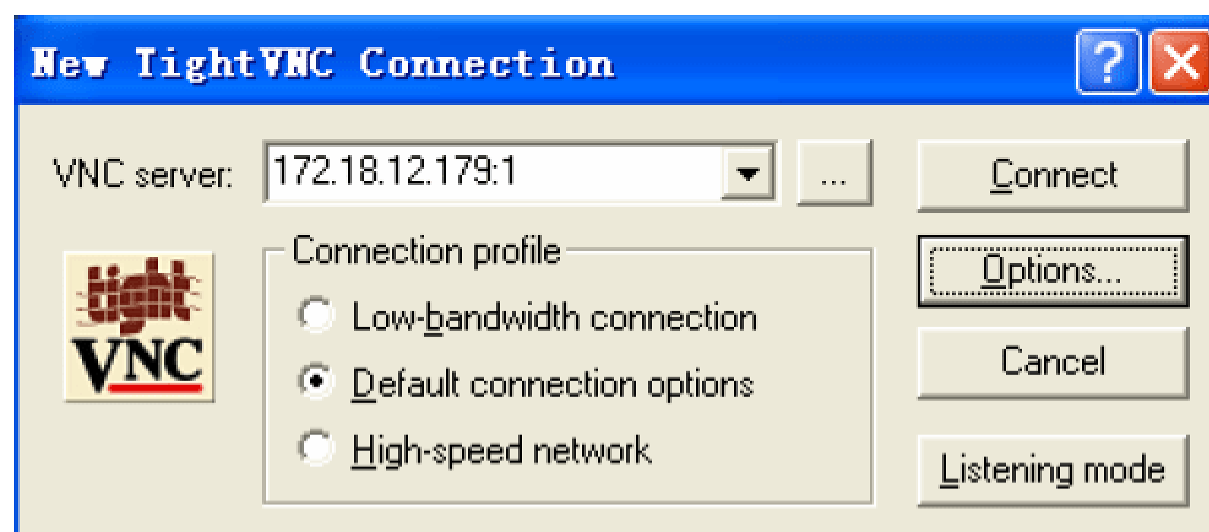


图 4-34　VNC Viewer 初始界面

❹ 单击图 4-34 中的 Connect 按钮，Tight VNC Viewer 尝试连接远程服务器。连接上后出现如图 4-35 所示的界面，根据前面配置，1 号桌面与 root 用户相关联，因此，需要输入 root 用户的密码。

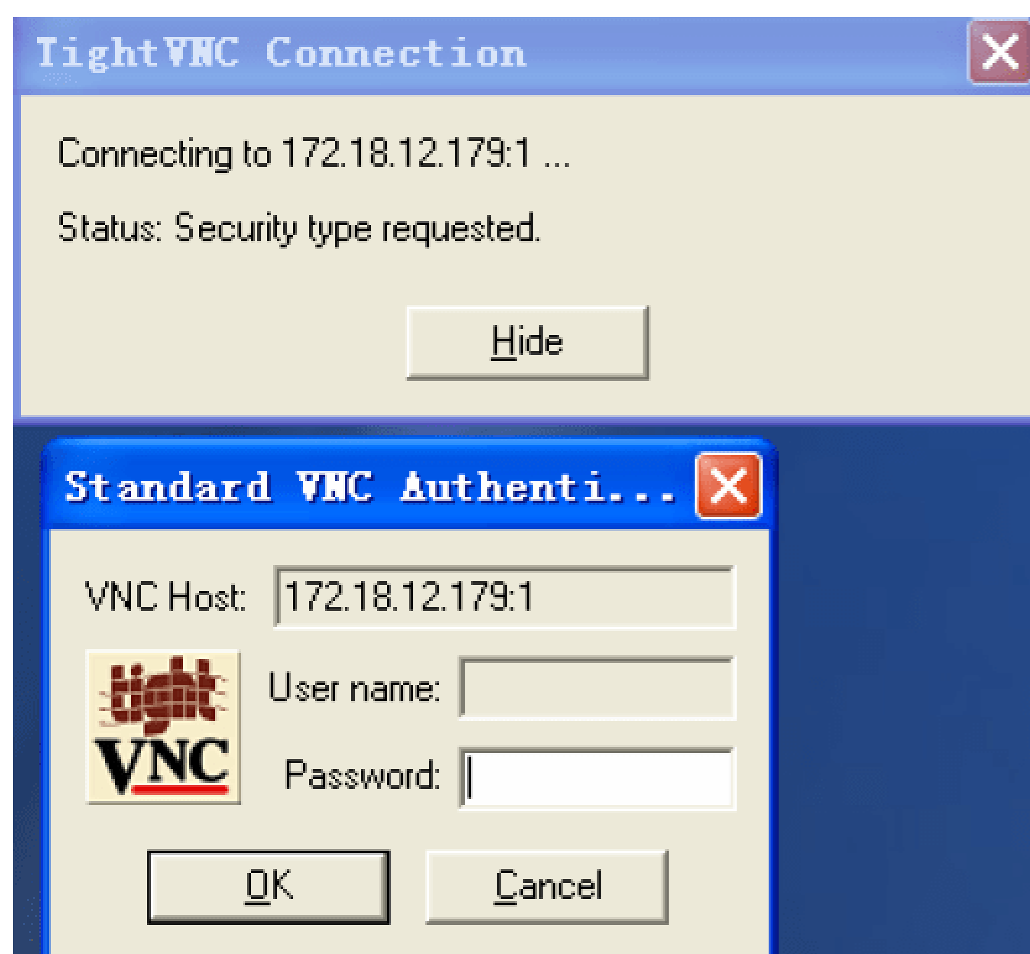


图 4-35　输入密码

❺ 正确输入密码，单击 OK 按钮，则出现 Linux 服务器的 X-Window 界面如图 4-36 所示。通过 VNC 我们就可以像坐在 Linux 服务器前面一样操纵它了。

❻ 在 X-Window 界面下打开终端，这时的操作就类似于你在 Windows 操作系统下操作了，不过 Linux 的 X-Window 界面不如 Windows 操作方便。假设 gftp 安装在/usr/local/gftp 目录下，进入该目录的/bin 目录，可以看到有 gftp 可执行文件，执行它就能打开图形化的 gftp 工具，启动 gftp 进程，如图 4-37 所示。

❼ gftp 界面如图 4-38 所示，它提供了与 Windows 系统下 CuteFTP 和 LeapFTP 等软件相类似的功能。正确输入 FTP 的 IP 地址，默认端口为 21，输入用户名和密码，就能登录指定的 FTP。图中左边窗口是本地 Linux 文件系统，右边窗口是 FTP 目录，利用中间的和按钮可以方便实现 FTP 的上传和下载功能。

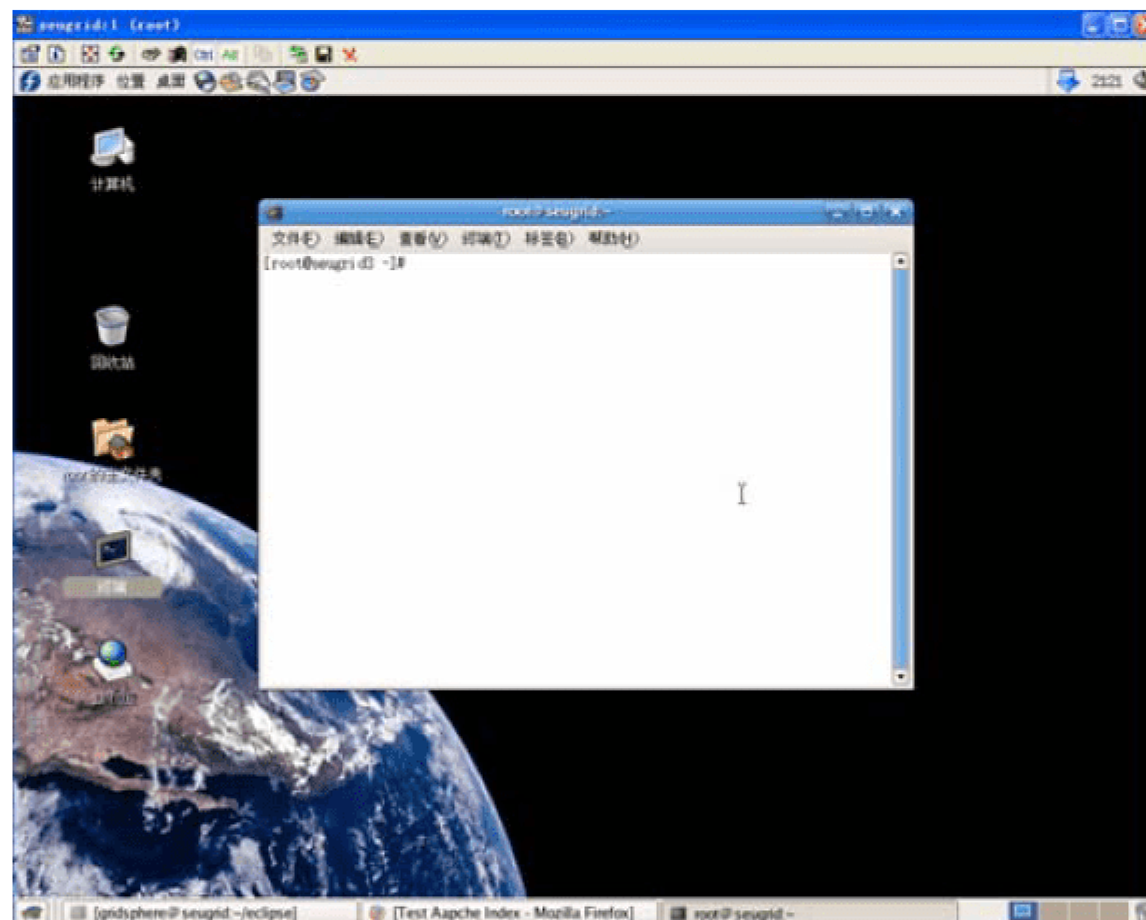


图 4-36　VNC Viewer 远程控制界面

```
[root@seugrid3 local]# cd gftp
[root@seugrid3 gftp]# ls
bin  man  share
[root@seugrid3 gftp]# cd bin
[root@seugrid3 bin]# ls
gftp  gftp-gtk  gftp-text
[root@seugrid3 bin]# ./gftp
```

图 4-37　启动 gftp 工具

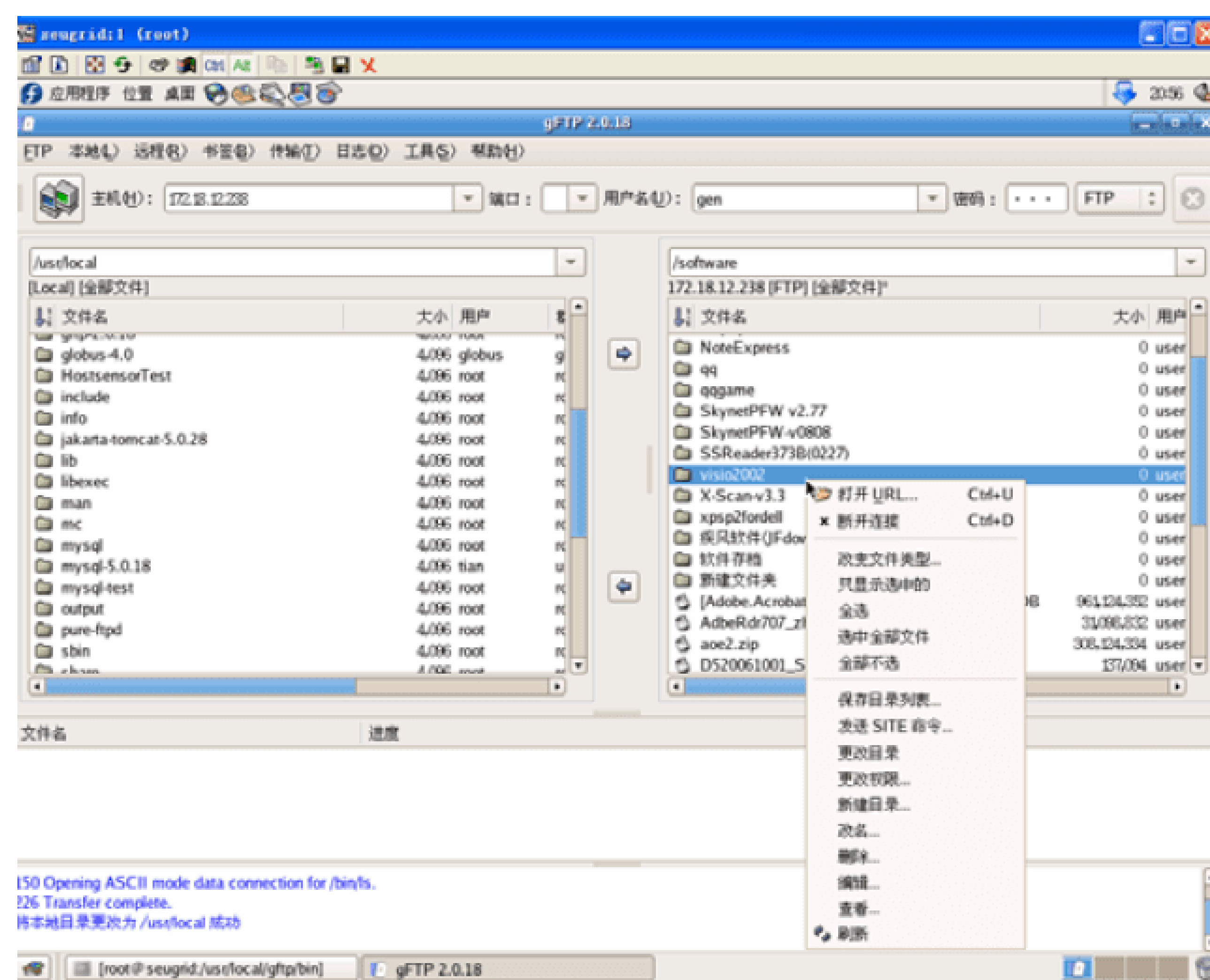


图 4-38　利用 gftp 下载软件

4.5　本 章 小 结

本章介绍了架设 Linux 服务器首先要解决的问题——远程控制问题，并详细介绍了目前 3 种远程控制解决方案：Telnet、SSH 服务和 VNC 服务。在介绍 3 种服务配置和应用的同时，简单介绍了非对称加密体系的基本原理、常见的服务器组网的拓扑结构、VNC 相对于传统图形化控制的优势等基础知识。SSH 提供了安全可靠的纯命令行方式控制服务器，与之相配合的 VNC，则提供了友好的图形化控制服务器的方式，这两种方法的结合使用完全可以解决远程控制方面的所有问题。

第 5 章　NFS 和 NIS 服务器的配置与应用

NFS(Network File System)最基本的思想就是让任意多的客户端与服务器共享一个文件系统，多用于 UNIX/Linux 所构建的分布式系统中。NIS(Network Information Service)服务用于在大型网络中统一集中管理多台 Linux 主机的用户的账号数据，NIS 服务器相当于黄页，当有用户要登录网络中的某台 Linux 主机时，该主机就会到 NIS 服务器上去查询该用户的账号和密码以进行验证。本章介绍 NFS 和 NIS 的原理、服务器端的配置方法及客户端的配置和测试等内容，最后结合 NFS 和 NIS 来实现对分布式系统用户账号和根目录的统一管理。

通过本章的学习，读者应掌握以下主要内容：

- ✧ NFS 的工作原理
- ✧ NFS 服务器端的配置
- ✧ NIS 的工作原理
- ✧ NIS 主从服务器的配置
- ✧ 结合 NFS 和 NIS 管理系统用户

5.1　NFS 服务简介

5.1.1　NFS 服务概述

NFS(Network File System)称为网络文件系统，是不同 UNIX/Linux 计算机之间通过网络实现文件共享的一种网络协议，最早是由 Sun 公司开发出来的。 NFS 使用方便，得到了广泛的认可，并被国际互联网工程组制定为 RFC1904、RFC1813 和 RFC3030 标准。

NFS 最大的功能就是可以透过网络，让不同的机器、不同的操作系统彼此分享个别的档案(share files)。所以，小型公司或机构常常利用 NFS 实现简单的文件服务器(file server)。这类似于 Windows 系统的文件资源共享，不过 NFS 是在 UNIX/Linux 系统下实现的。NFS 通过挂载服务器的共享目录来访问它们，当客户挂载一个远程目录时，该目录就成为本地目录树的一部分。许多 Sun 工作站都是无盘站，一台无盘站可以将一个远程的目录安装成它的根目录，它的整个文件系统完全由远程的服务器来支持。而有本地硬盘的工作站则可以将远程的目录挂载到本地目录树中，形成一个部分是本地、部分是远程的文件系统，对于在客户机上运行的程序来说，文件在本地还是在远程服务器上是没有关系的。

尽管 NFS 允许一台机器可以同时既是客户端又是服务器，但是为简单起见，假设客户端与服务器运行在不同的机器上。NFS 服务器的基本结构如图 5-1 所示，NFS 服务器将目

录/root/nfsshare 设置为共享目录，其他客户端主机可以将该共享目录挂载到本地系统的某个目录下，这个目录由客户端自己定义，不同客户端主机的挂载目录可以不相同，但是需要注意的是，每个目录不能挂载多个 NFS 服务器的共享目录。例如图中客户端 1 的挂载目录是/opt/mnt，而客户端 2 的挂载目录是/home/bob/mnt。挂载成功后就可以在挂载目录下看到与/root/nfsshare 完全一样的子目录和文件，只要 NFS 服务器设置相应的权限，客户端就可以像操作本地磁盘一样利用 ls、cd、rm、df 等命令对共享目录进行文件操作。

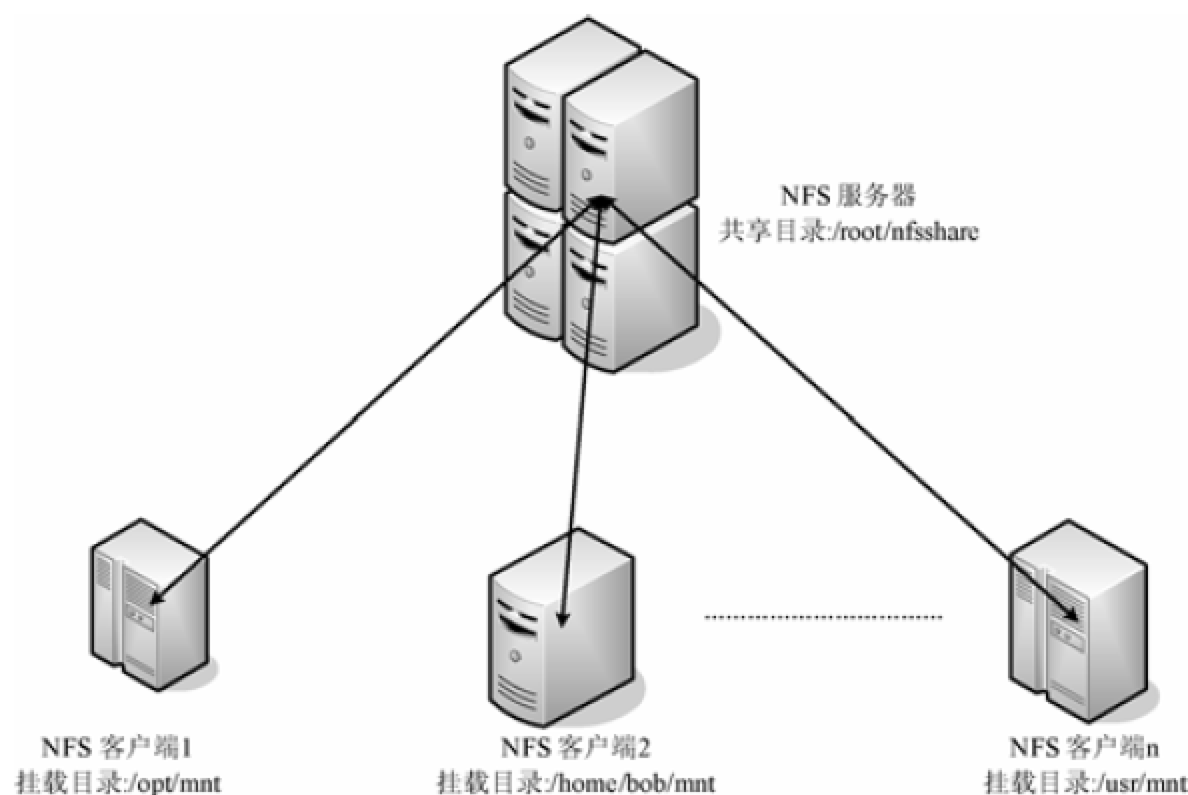


图 5-1　NFS 服务器与客户端挂载示意图

如果认为 NFS 仅仅可以用来架设文件服务器，那么 NFS 就显得大材小用了。在当今网络计算领域，因为大型机、中型机、小型机的发展已经受到了芯片设计的制约，所以分布式计算已经成为了高性能计算的主流。所谓分布式计算，就是联合多个计算结点(这些结点可以包括大型机、中型机、小型机、PC 机以及最为流行的刀片服务器)协同计算。目前美国正在研制 pero-bytes 的高性能计算机，它集成上万个刀片服务器进行计算，仅摆设这些刀片服务器的机架就足足有三个足球场那么大。在这种大规模的服务器管理中，NFS 就显示出了特殊的优势：不难设想，逐一安装这上万个刀片的应用程序需要花费多少的人力物力，那么利用 NFS 这些刀片服务器就可以直接将其应用程序挂载到本地，大大方便了资源的集中管理。本章 5.4 节将提及的网络信息服务 NIS(Network Information Service)则可以用来统一管理大规模服务器的用户和账号。因此，NFS 和 NIS 服务成为了大规模网络管理的两大利器。

5.1.2　NFS 协议的工作原理

既然 NFS 的一个目标就是支持异构系统，客户端与服务器能运行在不同的操作系统和不同的硬件上，这样就需要在客户端与服务器之间定义一个清晰的接口。NFS 定义两个客户端/服务器协议来达到上述目标。协议是由客户端向服务器发送请求，以及服务器返回给客户的响应组成的集合。协议是计算机网络的主线，也是分布式系统中的永恒主题，只有对网络协议有了深入全面的把握，才能真正理解计算机网络、理解分布式系统。

NFS 的第一条协议是关于目录挂载的，客户端将一个路径发给服务器，请求将该路径

所说明的目录挂载到它的目录树中，消息中不需要包含客户端的目录的挂载点的信息，因为服务器不需要关心挂载点。只要路径合法，并且该目录在共享目录集合中，服务器就将返回一个文件指针给客户端。这个指针惟一地确定了文件系统的类型、硬盘、该目录对应的 I 结点号以及安全信息，随后对该目录的读写操作就可以通过该文件指针来进行。

NFS 的第二条协议是关于目录和文件访问的，客户端通过向服务器发送消息，以操作目录或读写文件，也可以访问到文件的属性，如文件的模式、大小、最后一次修改的时间等。

NFS 协议是通过远程过程调用机制(Remote Procedure Call，RPC)来实现数据共享和传输的，RPC 定义了进程间网络通信的机制，屏蔽了底层通信协议的细节，允许客户端进程通过网络向远程服务器上的服务进程请求服务。NFS 服务器实际上就是一个 RPC 服务器，RPC 指定每个 NFS 功能所对应的端口号，并且回馈给客户端，让客户端可以连接到正确的端口上。那么 RPC 又是如何知道每个 NFS 的端口呢？原来当服务器在启动 NFS 时会随机取用多个端口，并主动向 RPC 注册，因此 RPC 可以知道每个端口对应的 NFS 功能，然后 RPC 又固定使用 111 号端口监听客户端的需求并回馈客户端正确的端口。

图 5-2 描述了 NFS 的工作原理，包括以下三个步骤。

客户端向服务器端的 RPC(端口 111)发出 NFS 文件存取功能的询问要求；

服务器端 RPC 找到对应的已注册的 NFS daemon 端口后，反馈给客户端；

客户端得到正确的端口后，就可以直接与 NFS daemon 通信了。

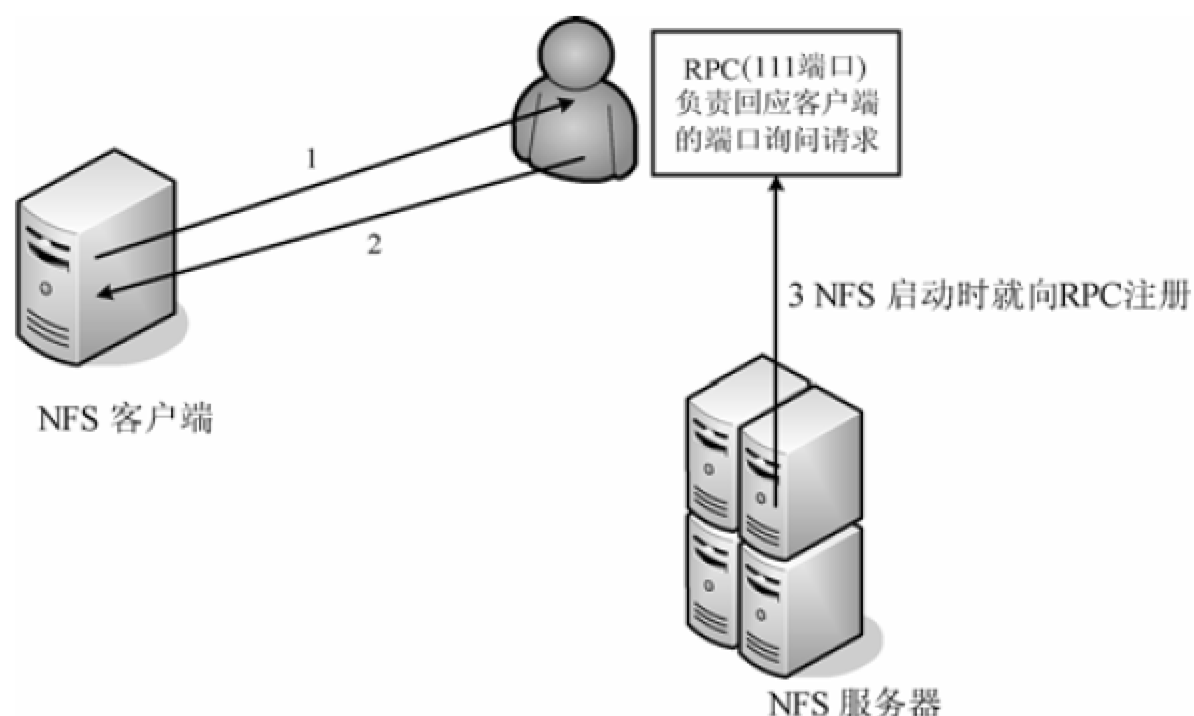


图 5-2 NFS 工作原理示意图

至于启动 NFS 服务，则至少需要启动以下三个系统守护进程：

✧ rpc.nfsd

  最基本的 NFS 守护进程，判别客户端用户是否能够登录服务器。

✧ rpc.mountd

  RPC 安装守护进程，主要功能是管理 NFS 的文件系统。当客户端成功通过 rpc.nfsd 进入 NFS 服务器后，在使用 NFS 服务器所提供的文件前，还必须通过文件使用权限的验证，rpc.mountd 通过读取 NFS 的配置文件/etc/exports 来管理客户端用户的权限。

✧ portmap

  该进程主要负责端口映射，当客户端试图连接并使用 RPC 服务器所提供的服务时，

portmap 将所管理的与服务器对应的端口号提供给客户端，即图 5-2 中的步骤 2 所描述的功能。

5.2　NFS 服务的配置

应用实例导航——A 公司架设文件服务器

※场景呈现

假设 A 公司准备架设一台简易的文件服务器，以便内部子网的资源共享和文件交流；并且该公司需要对外提供一部分文件资料，但是不允许外面 IP 的主机对文件系统进行写操作。网络拓扑如图 5-3 所示，主结点的外部 IP 为 211.65.63.146，局域网 IP 为 192.168.10.1，其他结点的 IP 从 192.168.0.2 开始分配，至 192.168.0.33 为止。

所架设的文件服务器的详细要求，描述如下。

(1) 在局域网内开放/home/users 目录，用于支持 NIS。

(2) 在局域网内开放/jogmgr 和/opt 目录，分别用于支持作业结果和应用程序。

(3) 对外开放/root/nfs 目录，使用属性为只读，除了提供网域内的工作站外，向外亦提供数据内容。

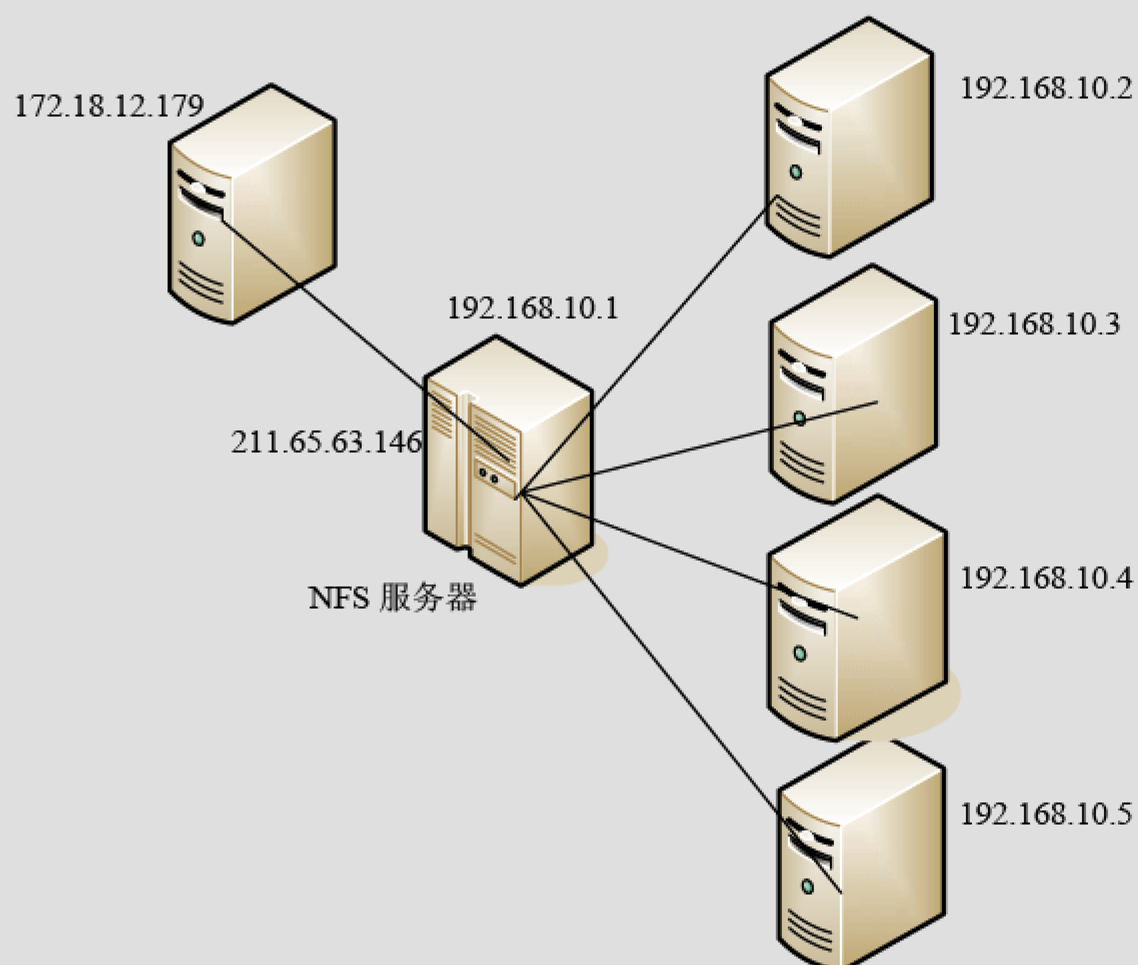


图 5-3　A 公司 NFS 服务器拓扑结构

※技术要领

(1) 配置 NFS 服务器端。

(3) 测试 NFS 服务器。

本实例的应用场景在实际工作中很常见，其文件服务器的配置要求也是非常基本的。下面将分别介绍 NFS 服务器端和客户端的配置以实现实例中的功能要求。

5.2.1 NFS 服务器端的配置

目前几乎所有 Linux 发行版都默认安装了 NFS 服务，因此无需再安装 NFS 服务。只是在配置使用 NFS 前，需要检查一下系统中是否已安装好 nfs-utils 和 portmap 这两个软件包。NFS 服务器端的配置相对简单，只需对/etc/exports 文件进行配置即可。

❶ Fedora 6.0 默认安装了 NFS 服务，因此可以省略安装过程，直接进入配置过程；在进行 NFS 配置前，可以使用下面命令检查系统是否已经安装 NFS 服务。

```
rpm -q nfs-utils portmap
```

如图 5-4 所示，系统当前已经安装了 NFS 所需要的 nfs-utils 和 portmap 两个包，并列出了这两个包的版本号。

```
[root@seugrid3 ~]#  rpm -q nfs-utils portmap
nfs-utils-1.0.8.rc2-4.FC5.2
portmap-4.0-65.2.2
[root@seugrid3 ~]# █
```

图 5-4　检查系统是否已经安装 NFS 服务

❷ /etc/exports 文件是 NFS 服务器的主要设定文件， exports 文件中的每一行提供了一个共享目录的设置，其命令格式为：

```
<共享目录>  [客户端1(参数1,参数2,…)] [客户端1(参数1,参数2,…)]
```

需要注意的是，共享目录为必选参数。

另外，格式中的共享目录和客户端之间、客户端与客户端之间都有空格符，但是客户端和参数之间不能有空格。

/etc/exports 配置文件的客户端指定可以有多种设定方法，它定义了网络中可以访问这个 NFS 服务器共享目录的计算机。/etc/exports 可以指定单个主机的 IP 地址或域名，也可以指定某个子网或域中的主机。

例如，本例要对局域网内部的所有主机开放/home/users、/jobmgr 和/opt 这三个目录，需要在 exports 文件添加如图 5-5 所示的三行语句，其中：

192.168.10.0/255.255.255.0 指定了范围在 192.168.10.1～192.168.10.255 的所有 IP;

rw 指定共享目录设置为可读可写；

no_root_squash 的意思是如果 root 用户进入 NFS 共享目录，他也具有 root 权限；

async 的意思是将数据先保存在内存缓冲区中，在恰当的时间才写入磁盘。

❸ 接下来，将/root/nfs 目录设置为对内对外都开放，需要在 exports 文件添加如图 5-6 中标注的语句，实际上语句：

```
/root/nfs *(ro)
```

就是设定将/root/nfs 目录对外开放的，并且权限仅可读。

```
[root@cgsp ~]# vi /etc/exports

#
/home/users     192.168.10.0/255.255.255.0(rw,insecure,no_root_squash,async)
/jobmgr         192.168.10.0/255.255.255.0(rw,insecure,no_root_squash,async)
/opt            192.168.10.0/255.255.255.0(rw,insecure,no_root_squash,async)
~
~
~
```

图 5-5 对局域网所有主机开放三个目录

```
[root@cgsp ~]# vi /etc/exports

#
/home/users     192.168.10.0/255.255.255.0(rw,insecure,no_root_squash,async)
/jobmgr         192.168.10.0/255.255.255.0(rw,insecure,no_root_squash,async)
/opt            192.168.10.0/255.255.255.0(rw,insecure,no_root_squash,async)
/root/nfs       192.168.10.0/255.255.255.0(rw,insecure,no_root_squash,async)    *(ro)
```

图 5-6 将/root/nfs 对外开放

❹ 与启动其他 Linux 服务不同的是，在启动 NFS 服务前，首先要启动 portmap 服务，利用下面两条命令即可完成，结果如图 5-7 所示。

```
/etc/init.d/portmap start
/etc/init.d/nfs start
```

启动 NFS 命令与启动其他 Linux 系统服务器类似，那么其他重启、停止、观察 NFS 服务状态等命令只需改变命令之后的参数即可，在此不再详细论述。

```
[root@seugrid3 etc]# /etc/init.d/portmap start
启动 portmap:                                        [确定]
[root@seugrid3 etc]# /etc/init.d/nfs start
启动 NFS 服务:                                       [确定]
关掉 NFS 配额:                                       [确定]
启动 NFS 守护进程:                                   [确定]
启动 NFS mountd:                                     [确定]
```

图 5-7 启动 NFS 服务

❺ 如果仅仅改变/etc/exports 文件内容，则无需重启 NFS，只需输入下面命令让/etc/exports 生效即可，如图 5-8 所示。

```
[root@cgsp ~]# exportfs -arv
exporting 192.168.10.0/255.255.255.0:/home/users
exporting 192.168.10.0/255.255.255.0:/root/nfs
exporting 192.168.10.0/255.255.255.0:/jobmgr
exporting 192.168.10.0/255.255.255.0:/opt
exporting *:/root/nfs
[root@cgsp ~]#
```

图 5-8 重新读取/etc/exports 配置

```
exportfs -arv
```

exportfs 命令可以带四个可选参数，如表 5-1 所示。

表 5-1　exportfs 命令参数

exportfs 命令参数	描　述
-a	输出在/etc/exports 中设置的所有目录
-r	重新读取/etc/exports 文件中的设置，并使设置立即生效，但不需要重启 NFS 服务
-u	停止某一共享目录
-v	回显所设置的共享目录

点评与拓展：只要理解/etc/exports 文件的含义和 exportfs 命令的使用方法，就不难配置 NFS 服务器了。另外，portmap 使 Linux 系统用于支持 RPC 机制，所有用到 RPC 机制的服务需要首先启动 portmap 服务。

5.2.2　NFS 客户端的配置和测试

NFS 服务器端配置好后，还需要对客户端作一点简单的配置，并且测试所架设的 NFS 服务器。

❶ 客户端同样需要启动 portmap 服务，该服务一般默认为启动，可以输入下面命令确认：

```
ps -aux | grep portmap
```

图 5-9 标注部分显示 portmap 服务已经成功启动，如果没有启动则使用下面命令启动：

```
/etc/init.d/portmap start
```

```
[root@seugrid3 ~]# ps -aux | grep portmap
Warning: bad syntax, perhaps a bogus '-'? See /usr/share/doc/procps-3.2.6/FAQ
rpc       1563  0.0  0.0   1728   616 ?       Ss   Mar27   0:00 portmap
root      3867  0.0  0.0   5164   740 pts/2   S+   16:48   0:00 grep portmap
[root@seugrid3 ~]#
```

图 5-9　确认 portmap 服务是否启动

❷ 启动客户端的 portmap 服务后，我们开始测试 NFS 服务器，首先在局域网内的主机上测试。SSH 登录局域网内 IP 为 192.168.10.2 这台主机，输入 showmount 命令可以看到 NFS 服务器有哪些共享目录、这些权限是什么，用到的命令分列于下：

```
ssh 192.168.10.2
showmount –e 192.168.10.1
```

结果如图 5-10 所示，可以看到 showmount 命令返回的结果与我们在 NFS 服务器端的/etc/exports 中所设置的内容一致。

❸ 因为/opt 目录对局域网内主机都开放，这里以挂载/opt 为例说明挂载的过程与效果。输入下面命令挂载某个共享目录：

```
mount -t nfs 192.168.10.1:/opt /opt
```

该命令说明将主机 192.168.10.1 的共享目录/opt 挂载到本机的/opt 下，本机的/opt 目录称为挂载点(mount point)。挂载成功后，输入 ls /opt 命令，就会列出 NFS 服务器的/opt 目录内容；输入 df 命令会看到除了本机的磁盘外，多了一个 192.168.10.1:/opt，如图 5-11 所示。

```
[root@cgsp ~]# ssh 192.168.10.2
Last login: Wed Oct 10 21:29:22 2007 from node1ib.job
-bash: /jobmgr/conf/profile.lsf: No such file or directory
[root@node2ib ~]# showmount -e 192.168.10.1
Export list for 192.168.10.1:
/opt        192.168.10.0/255.255.255.0
/jobmgr     192.168.10.0/255.255.255.0
/root/nfs   (everyone)
/home/users 192.168.10.0/255.255.255.0
[root@node2ib ~]#
```

图 5-10　显示 NFS 共享目录及权限

```
[root@node2ib /]# mount -t nfs 192.168.10.1:/opt /opt
[root@node2ib /]# ls /opt
emc  hpl  mpich  TSprocess  TSprocess1.0.tar.gz
[root@node2ib /]# df
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/mapper/VolGroup00-LogVol00
                      69893592   7286392  59056788  11% /
/dev/sda1               101086     18291     77576  20% /boot
none                   1027812         0   1027812   0% /dev/shm
192.168.10.1:/opt      1032096    539264    440416  56% /opt
```

图 5-11　挂载/opt 目录

❹ 那么如何卸载共享目录呢？Linux 系统提供了 umount 来实现，如下：

```
umount /opt
```

结果如图 5-12 所示，这时再输入 df 查看磁盘命令时，就发现少了 192.168.10.1:/opt 这项，说明该共享目录已经被成功卸载。

```
[root@node2ib /]# umount /opt
[root@node2ib /]# df
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/mapper/VolGroup00-LogVol00
                      69893592   7286388  59056792  11% /
/dev/sda1               101086     18291     77576  20% /boot
none                   1027812         0   1027812   0% /dev/shm
```

图 5-12　卸载/opt 目录

❺ 上面两步测试了内部局域网的主机，实例中要求将/root/nfs 开放给外部主机，在此测试外部主机是否能够挂载/root/nfs 目录。
登录 IP 为 172.18.12.179 的主机，输入下面命令查看 NFS 服务器所开放的共享目录，注意命令中用的是 NFS 服务器的外部 IP 地址，由拓扑图知，外部 IP 地址为 211.65.63.146。

```
showmount -e 211.65.63.146
```

图 5-13 显示了 NFS 服务器提供的共享目录，与内部主机的显示结果一样。

同样可以利用 mount 命令挂载/root/nfs 目录，如图 5-13 所示。然后，使用 df 命令查看 mount 的结果，可以发现 172.18.12.179 这台外部主机多了 211.65.63.146:/root/nfs 共享目录，说明该 NFS 服务器在外部主机上也可以挂载成功，所使用到的两条命令分列如下：

```
mount -t nfs 211.65.63.146:/root/nfs /root/nfs
df
```

其中：/root/nfs 为 IP 为 172.18.12.179 的主机的挂载点。

卸载命令与内部主机一样：

```
umount /root/nfs
```

```
[root@seugrid3 ~]# showmount -e 211.65.63.146
Export list for 211.65.63.146:
/opt         192.168.10.0/255.255.255.0
/jobmgr      192.168.10.0/255.255.255.0
/root/nfs    (everyone)
/home/users 192.168.10.0/255.255.255.0
[root@seugrid3 ~]#  mount -t nfs 211.65.63.146:/root/nfs /root/nfs
[root@seugrid3 ~]# df
文件系统               1K-块        已用      可用 已用% 挂载点
/dev/mapper/VolGroup00-LogVol00
                      66784856  16426848  46910816  26% /
/dev/cciss/c0d0p1       101086     10090     85777  11% /boot
tmpfs                   516240         0    516240   0% /dev/shm
211.65.63.146:/root/nfs
                       1032096    539264    440384  56% /root/nfs
[root@seugrid3 ~]#
```

图 5-13　外部主机挂载/root/nfs

📖**点评与拓展**：需要注意的是，内部局域网挂载 NFS 服务器的共享目录是通过服务器的内部 IP 地址，外部主机则是通过 NFS 服务器的外部 IP 地址，正如场景中网络拓扑图所展示的一样。

5.3　主从架构下的 NFS 服务

应用实例导航——为主从架构网络配置 NFS 服务

※场景呈现

为了满足高性能计算的需求，某机构购置了 32 台刀片服务器，用于并行计算，这些刀片服务器由一台主服务器统一控制，它们具有同样的操作系统和文件系统，通过高速链路耦合。为了统一高效地管理这些刀片服务器的文件系统，网络管理员不希望一台一台登录从结点挂载主服务器的共享目录，而希望通过主控结点统一远程为这些刀片服务器挂载或卸载共享目录。

要求结合 NFS 配置、shell 脚本的编写等知识实现在主控结点端的统一挂载和卸载共享目录。网络拓扑如图 5-14 所示，主结点的局域网 IP 为 192.168.10.1，主机名为 node1ib。其他结点的 IP 从 192.168.0.2 开始分配，至 192.168.0.33 为止；主机名从 node2ib 开始分配，到 node33ib 为止。32 个结点的主机名和 IP 地址的对应关系已经写入到主服务器及所有子结点的/etc/hosts 文件中。

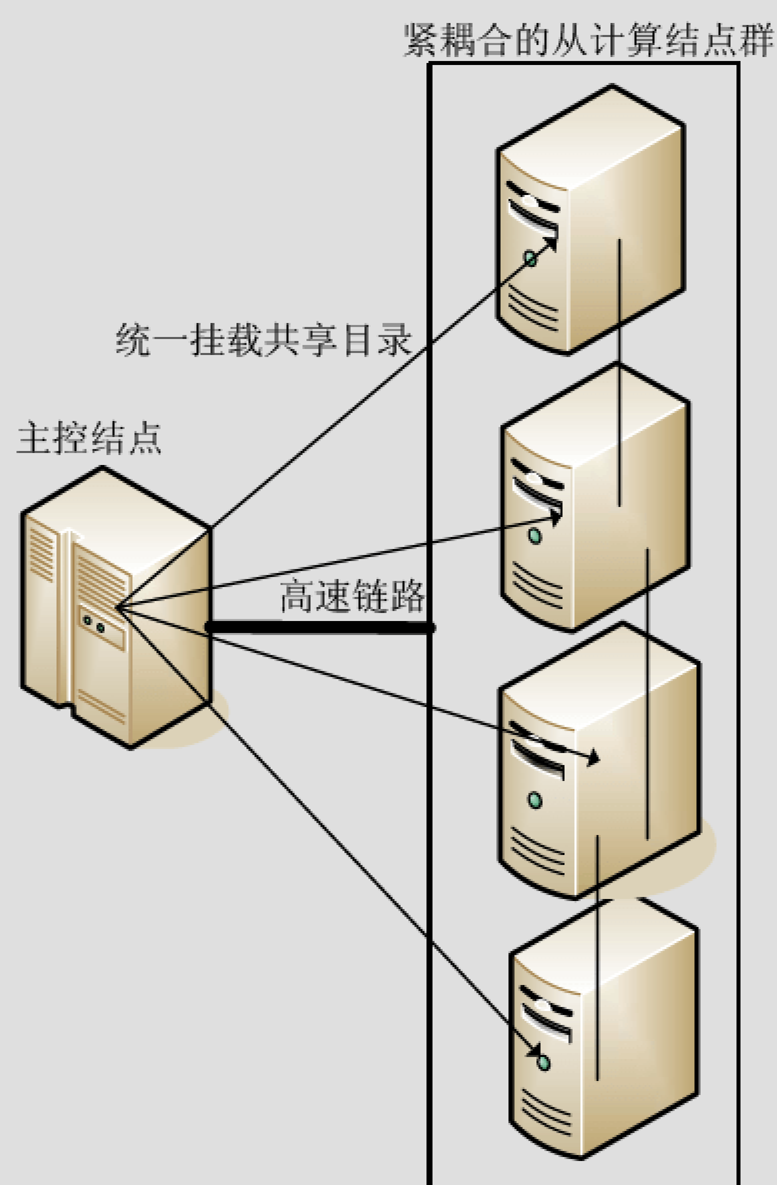


图 5-14　主从架构拓扑示意图

※技术要领

(1) 理解何谓主从架构，NFS 在主从架构下的存在意义。

(2) 实现在主控**结点**统一挂载和卸载共享目录。

5.3.1　主从架构下 NFS 服务的需求

早期的计算机一直是庞大昂贵的设备，即便是小型机也要花费数万美元，这使得大多数机构只有为数不多的几台机器，而且由于这些机器间缺乏互联的手段，通常只能各自独立地工作。从 20 世纪 80 年代开始，微型计算机的功能变得越来越强大，由 8 位的微机发展到了普遍的 32 位，现在甚至 64 位的微机也变得非常普遍，它们具有早期大型机的计算能力，但是价格却低得多。正是这个经济上的因素促使了计算机系统由集中式发展到分布式，一大群微机由高速局域网互联所组成的分布式系统不仅比集中式系统有更好的性价格，而且具有任何价位上的大型机都无法达到的性能。其他的很多优点也使得分布式计算系统变得越来越流行。我们总结了分布式系统的几大优点，如表 5-2 所示。

表 5-2 分布式系统的优点

优　点	描　述
高性价比	微处理器通过高速局域网的互联提供比大型机更好的性价比
速度	分布式系统提供比大型机更强的计算能力
可靠性	当某些微机崩溃时，整个系统仍然能够正常运转
可扩展性	可以随时增加微机以增强计算能力

计算机主从架构(master/slave architecture)是分布式系统中最普遍的体系架构之一，其应用也变得越来越广泛。那么何谓主从架构呢？就是说，主控结点主机通过高速链路连接各从结点主机，并对其具有绝对的控制权，包括其文件系统的构建、计算任务的分配和回收等；而各从结点由同构的 Linux 服务器形成，各结点间具有相同的文件系统、系统软件以及系统用户，可以并行完成计算任务。在这种体系结构下，NFS 就大有用武之地了，因为所有的从结点主机需要有同样的应用程序目录或作业目录，而这可以通过挂载主控结点上的相应共享目录来实现。

但还有一个问题是，当从结点规模很大时，网络管理员也不愿意一台台地登录每个从服务器去挂载主控结点的共享目录，这需要耗费大量的精力，而希望直接在主控结点上统一为所有的从结点挂载共享目录，这与 5.2 节“NFS 服务的配置”所介绍的 NFS 服务挂载方式就有所不同了。本节将介绍这种从 NFS 服务器端统一挂载客户端共享目录，并具备良好扩展性的方法。目前，随着集群技术(cluster)的不断成熟，集群产品化和产业化的趋势也越来越明显，集群就是利用网络进行连接的多台计算机的集合，这些连接在一起的计算机使用起来像一个单一的一体计算资源。集群的操作系统通常使用的是 UNIX 或 Linux，体系架构一般采用的是主从架构，因此，NFS 服务器端统一挂载客户端共享目录的方法被广泛地使用到了集群中。

5.3.2 NFS 服务器端统一控制目录挂载

本节介绍的方法，需要在 SSH 无密码登录调试通过的基础上实现，也就是说主控结点和从**结点**两两之间都能够无密码登录，具体的 SSH 无密码登录的原理及配置方法请参见第 4.3 节“配置 SSH 无密码登录”的介绍。一旦 SSH 无密码登录成功后，主控结点就可以通过 SSH 协议远程执行命令，当需要在 NFS 服务器端统一控制共享目录挂载时只需远程执行挂载命令即可实现。由于通常对主从架构下的从属结点统一编号，因此可以将 SSH 协议远程执行命令封装成普遍适用的脚本以方便用户对从属结点的管理。下面介绍远程执行命令的脚本及实现 NFS 服务器端统一控制共享目录挂载的方法。

1. 在网络配置方面，为了方便管理，通常需要为从属结点统一设定主机名，假设本例的主控结点的 IP 为 192.168.10.1，主机名为 node1ib；那么从属结点的 IP 就可以从 192.168.10.2 开始分配、主机名从 node2ib 开始分配；第 i 个 Slave 主机 IP 为 192.168.10.i+1，主机名为 node(i+1)ib。我们将局域网 IP 和主机名的对应关系写入/etc/hosts 文件，如图 5-15 所示，这里我们写出前 4 台主机做示例，实际上，应该是有几台从属结点就写几行 IP 与主机名的对应关系。

```
[root@cgsp ~]# more /etc/hosts
        Do       not      remove  the      following       line,
# that require network functionality will fail.
127.0.0.1        localhost.localdomain    localhost
211.65.63.146    cgsp.job        cgsp     cgsp.local
192.168.10.1     node1ib.job      node1ib node1ib.local
192.168.10.2     node2ib.job      node2ib node2ib.local
192.168.10.3     node3ib.job      node3ib node3ib.local
192.168.10.4     node4ib.job      node4ib node4ib.local
```

图 5-15　/etc/hosts 文件配置

❷ 通过 SSH 协议远程执行命令可以通过层层嵌套的三个脚本来实现，这三个脚本的名字分别为 hostgen、sshrrun 和 sshrrunon。其中 hostgen 用于产生 Slave 主机名，这与上面的编号方式有关，上一步的编号使得从属结点主机名的区别仅在于中间数字的区别，因此加上前缀和后缀就形成了完整的主机名，如图 5-16 所示；sshrrun 在无密码登录的基础上，通过 SSH 协议实现远程执行命令；sshrrunon 封装了前两个脚本，提供对外接口，输入参数是需要远程执行的命令，以及起始主机编号和终止主机编号，分别作为 sshrrun 和 hostgen 脚本的输入参数。用户最终要使用的仅仅是 sshrrunon 脚本，而无需调用其他的两个脚本。

本书附带的光盘提供了这三个脚本，读者只需将其复制到 Linux 的某个目录下，本例将它们复制到/root/nfs 目录中，将权限都改成可执行，这样就可以直接调用 sshrrunon 脚本。在此，不详细解释这三个脚本的语义，因为语义与 shell 编程规范相关，其基本知识在 1.3.3 节“shell 脚本”中作过简单介绍，在此直接讲述如何利用这几个脚本实现场景的要求。图 5-16、图 5-17 和图 5-18 分别显示了 hostgen、sshrrun 和 sshrrunon 三个 shell 脚本的代码。

```
[root@cgsp nfs]# ls
hostgen  sshrrun  sshrrunon
[root@cgsp nfs]# more hostgen
#!/bin/bash
if [ "$#" -eq "2" ]; then
    start=$1
    end=$2

    while [ "$start" -le "$end" ]
    do
        echo node${start}ib
        start=`expr $start + 1`
    done
fi
[root@cgsp nfs]#
```

图 5-16　hostgen 脚本

```
[root@cgsp admin]# more sshrrun
#!/bin/bash
if [ "$#" -ne "2" ]; then
   echo "USAGE sshrrun host cmd_string"
   exit 0
fi
/usr/bin/ssh $1 "$2"
```

图 5-17　sshrrun 脚本

❸ 进入包含上述三个脚本的目录，本例是进入/root/nfs 目录，使用下面命令将/opt 挂载到第 2

个结点和第 3 个结点，即 node2ib 和 node3ib 这两个结点，结果如图 5-19 所示。屏幕回显在哪个结点运行哪条命令，如果执行不成功回显错误信息。

```
sshrrunon "mount -t nfs 192.168.10.1:/opt /opt" 2 3
```

引号里面的是所要远程执行的命令，然后是起始结点号和终止结点号，上面命令的意思是在 2 号结点到 3 号结点上执行 mount -t nfs 192.168.10.1:/opt /opt 这条命令。读者一般只要知道 sshrrunon 这个脚本的用法即可。

如果需要挂载 node15ib～node33ib 范围内的所有结点，则命令可扩展为：

```
sshrrunon "mount -t nfs 192.168.10.1:/opt /opt" 15 33
```

当然还可以通过 sshrrunon "df" 2 3 这条命令查看 node2ib 和 node3ib 这两个结点上的挂载情况，结果如图 5-17 所示。

```
[root@cgsp admin]# more sshrrunon
#!/bin/bash
if [ "$#" -ne "3" ]; then
   echo "USAGE: sshrrunon cmd_string from_host end_host"
   exit 0
fi
for i in `hostgen $2 $3`
do
    echo "---------------sshrun $1 on $i-----------------"
    sshrrun $i "$1"
done
[root@cgsp admin]#
```

图 5-18　sshrrunon 脚本

```
[root@cgsp nfs]# sshrrunon "mount -t nfs 192.168.10.1:/opt /opt" 2 3
---------------sshrun mount -t nfs 192.168.10.1:/opt /opt on node2ib.job---------
---------------sshrun mount -t nfs 192.168.10.1:/opt /opt on node3ib.job---------
[root@cgsp nfs]# sshrrunon "df" 2 3
---------------sshrun df on node2ib.job-----------------
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/mapper/VolGroup00-LogVol00
                      69893592   7286660  59056520  11% /
/dev/sda1               101086     18291     77576  20% /boot
none                   1027812         0   1027812   0% /dev/shm
192.168.10.1:/opt      1032096    539264    440384  56% /opt
---------------sshrun df on node3ib.job-----------------
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/mapper/VolGroup00-LogVol00
                      69893592   7220504  59122676  11% /
/dev/sda1               101086     18286     77581  20% /boot
none                   1027816         0   1027816   0% /dev/shm
192.168.10.1:/opt      1032096    539264    440384  56% /opt
[root@cgsp nfs]#
```

图 5-19　远程挂载共享目录

📖**点评与拓展**：这三个脚本很实用，稍作修改可以适用于其他主从结构的网络，能够远程执行一切 shell 命令。脚本的写法遵循 shell 编程规范，读者通过阅读相关 shell 编程介绍，就不难掌握。

5.4　NIS 服务简介

5.4.1　NIS 服务概述

NIS(Network Information Service)是一种能够集中管理网络中所有主机账号信息的服务器软件。一个大规模网络中，往往有多部 Linux 主机，如果每台主机都要重复设定相同的用户账号和密码，可想而知数据是相当冗余的。如果有一台主服务器专门管理所有主机的账号，当用户登录其他主机时，就到这台主服务器来请求相关的用户账号密码等信息，这样就能极大地避免重复设定用户账号信息的工作，节省主机空间资源，提高使用效率。此外，若要增加、修改或删除用户账号信息，也只要到该主服务器上处理就可以了。

我们知道，一台 Linux 主机的功能要尽可能单一，最好只提供一项专门服务，这样，一方面可以充分利用系统资源，另一方面系统发生问题时可以比较容易地找到问题所在。一个大型网络中，往往设有多台提供专门服务的 Linux 主机，如有的专门提供 Web 服务，有的专门提供 FTP 服务，有的专门提供 Mail 服务，等等。而所有这些 Linux 主机的用户账号和密码都是一样的，如果该网络上用户数量很大的话，那么每台 Linux 主机上用户账号管理这方面的工作量还是非常可观的。NIS 则提供了专门管理用户账号和密码的功能。NIS 主机是一台专门管理账号信息的主机，当其他 Linux 主机有用户要登录时，就必须到这台专门管理用户账号信息的主机来查询用户账号和密码。如果要管理所有 Linux 主机的账号信息，也只要到这台主机上去设定即可。

5.4.2　NIS 服务的工作流程

在一个大型网络中，所有的 Linux 主机都访问同一台 NIS 服务器请求用户数据，这可能会造成 NIS 服务器负载过大。如果这单独的一台 NIS 服务器出现单点失效，用户将无法登录其他的 Linux 主机。所以，在大型网络环境中，NIS 服务器一般采用 master/slave(主/从服务器)架构。主 NIS 服务器提供系统管理员制作的数据库，而从 NIS 服务器则取得来自主 NIS 服务器的数据，并提供给其他客户端查询。因而主从服务器的用户数据是同步的。客户端可以向整个网络广播用户数据的请求，主从 NIS 服务器都可以应答。这样就分担了 NIS 服务器的负载，也可以有效避免因 NIS 服务器单点失效导致的用户无法登录。需要说明的是，这里说的主从服务器尽管也称为 master/slave 架构，但是与 5.3.1 节中所介绍的主从架构略有区别，主 NIS 服务器只是在 NIS 服务方面对从 NIS 服务器有控制权，实际上，从 NIS 服务器可以理解为主 NIS 服务器用户数据的备用服务器。第 7 章“DNS 服务器的配置与架设”将要讲述的主从 DNS 服务器，也可以将从 DNS 服务器理解为主 DNS 服务器的备用服务器。

那么 NIS 服务的工作过程就包括了三个角色：NIS 主服务器端，负责将用户账户信息制成数据库，响应 NIS 客户端的查询请求，并提供给从服务器端来更新；NIS 从服务器端，

以主服务器端的数据库为自身数据库来源，响应 NIS 客户端的查询请求；NIS 客户端，向主服务器或从服务器请求登录用户的验证数据。

整个 NIS 的工作流程如图 5-20 所示，具体描述如下。

NIS 服务器端(master/slave)的工作流程：

NIS 主服务器端先将所有系统用户账号密码相关信息制作成数据库；

NIS 主服务器端可主动通知 NIS 从服务器端来更新，即以“推送(PUSH)”的方式更新 NIS 从服务器端的数据库；

NIS 从服务器也可主动要求从 NIS 主服务器获取数据库，即以“拉(PULL)”的方式更新 NIS 从服务器端的数据库；

账号密码有变动时，需要重新制作数据库，并使主从服务器端保持同步。

NIS 客户端的工作流程：

当有用户登录 NIS 客户端时，首先查询本机的/etc/passwd、/etc/shadow 等文件，判断是否是本机的系统用户；

如果在本机找不到相关的账号信息，即断定该登录不是本机的系统用户，就开始向整个 NIS 网络广播查询，判断是否是 NIS 用户；

每台 NIS 服务器(master/slave)都可以响应客户端的请求，先响应者优先。

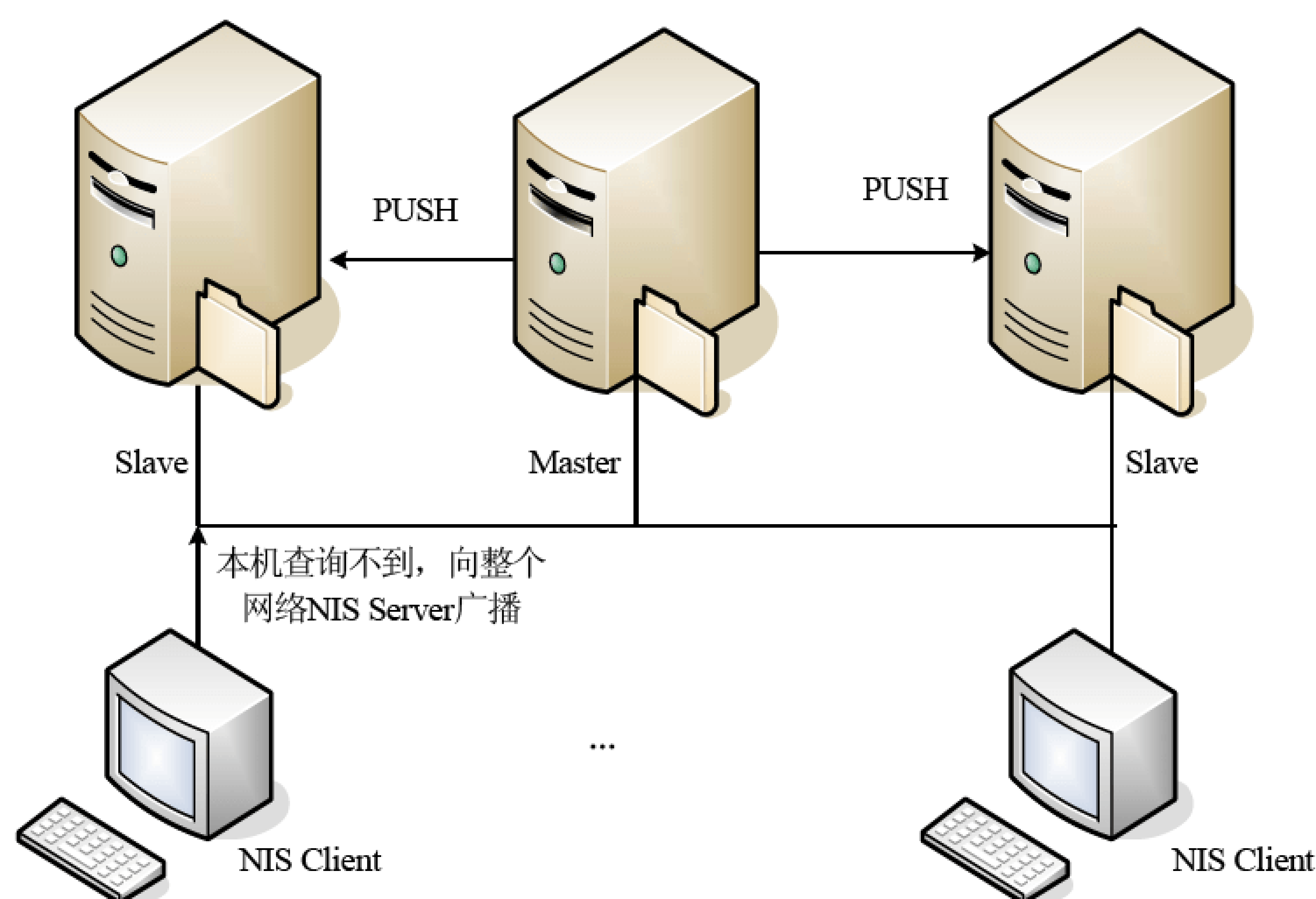


图 5-20　NIS 工作流程图

从上述流程可以看出，NIS 客户端是先对本机上的账号信息进行查询，查不到才到 NIS 服务器上查询。因此，当 NIS 客户端本身就有很多一般用户的账号时，就可能会跟 NIS 服务器提供的账号产生一定程度的差异。所以，一般在这种情况下 NIS 客户端或 NIS 从服务器会主动删除本机自身的一般用户账号，仅保留系统所需的 root 及少量其他账号。 这样，一般用户就都可以由 NIS 服务器进行控制了。

5.5　NIS 服务的配置

应用实例导航——架设主从 NIS 服务器

※场景呈现

A 公司需要架设 NIS 服务器，为了均衡查询负载并避免单点失效，NIS 服务器采用了主从结构，即有一个主服务器和一个从服务器，它们之间使用推送(PUSH)方式保持 NIS 数据库同步。

网络拓扑如图 5-21 所示。NIS 域名为 job，主服务器局域网 IP 为 192.168.10.1，从服务器局域网 IP 为 192.168.10.2，其他主机都为 NIS 客户机，IP 范围为 192.168.10.3～192.168.10.254。

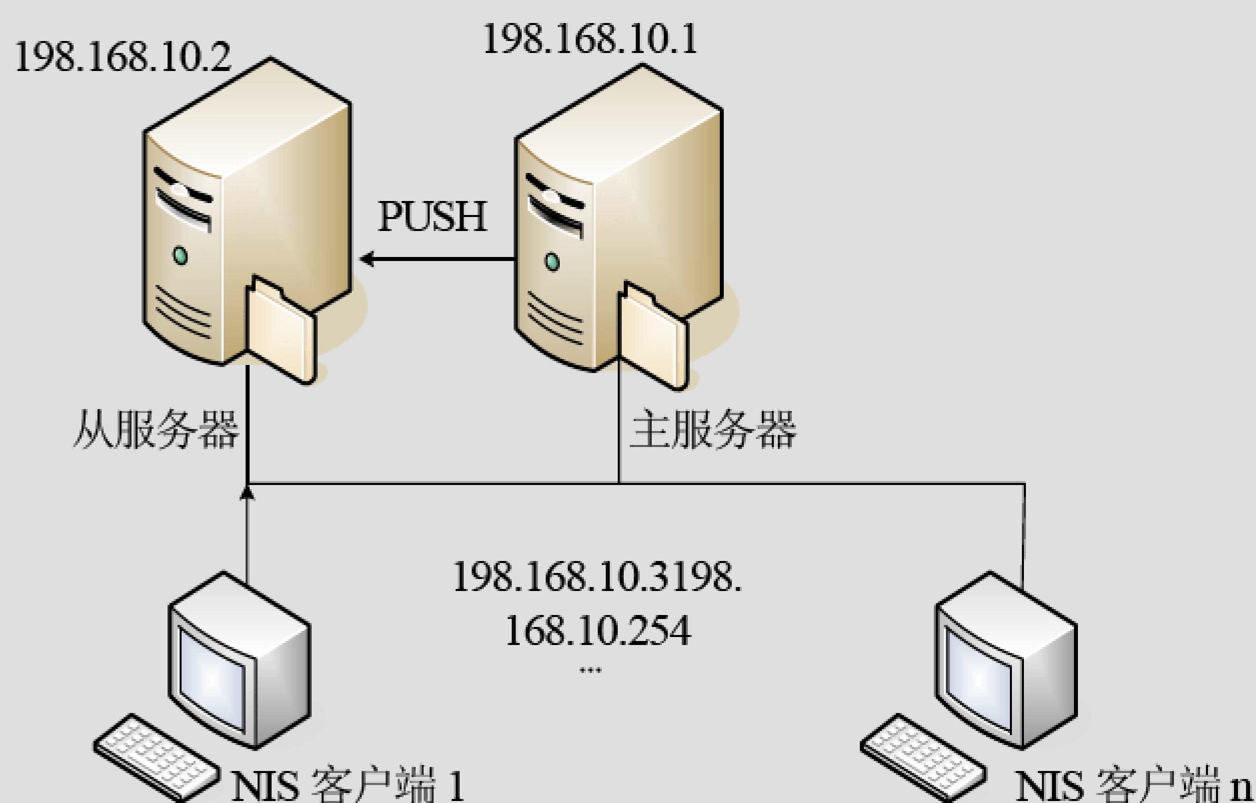


图 5-21　NIS 服务器拓扑结构

※技术要领

(1) 掌握 NIS 主服务器端的配置。
(2) 掌握 NIS 从服务器端的配置。
(3) 掌握 NIS 客户端的配置。
(4) 掌握如何测试 NIS 客户端。

本实例中 NIS 采用主从架构，因为有主从两个服务器，所以服务器端需要配置两次。本节从 NIS 主服务器端的配置工作开始讲起，NIS 从服务器端的配置与主服务器端类似。在服务器端配置成功后，方可对 NIS 客户端进行配置。

5.5.1 NIS 主服务器端的配置

NIS 拓扑中包含了三类不同功能的主机：主服务器(master)、从服务器(slave)以及客户端(client)。本节介绍 NIS 主服务器端主机的相关配置。

❶ 由于 NIS 服务是在某个域内工作的，因此首先要设定 NIS 服务的域名，域名的设置要与 /etc/hosts 文件中的设定一致。设定域名可以使用 nisdomainname 或 ypdomainname 两个命令实现，命令用法分列于下。

```
nisdomainname job
```

或

```
ypdomainname job
```

以上两条命令都是将 NIS 服务的域名设为 job，并且与/etc/hosts 中的一致，如图 5-22 所示。另外，nisdomainname 或 ypdomainname 如果后面不带参数，则显示已经设定好的域名。Linux 系统启动时会自动执行 rc.local 中的所有命令，所以将 ypdomainname job 命令写入系统的/etc/rc.local 文件，这样系统启动时就会自动设定 NIS 的域名，如图 5-23 所示。

```
[root@cgsp ~]# nisdomainname job
[root@cgsp ~]# nisdomainname
job
[root@cgsp ~]# more /etc/hosts
        Do       not      remove  the      following       line,    or
# that require network functionality will fail.
127.0.0.1        localhost.localdomain    localhost
211.65.63.146    cgsp.job        cgsp     cgsp.local
```

图 5-22　设定 NIS 域名

```
[root@cgsp ~]# more /etc/rc.local
#!/bin/sh
#
# This script will be executed *after* all the other init scripts.
# You can put your own initialization stuff in here if you don't
# want to do the full Sys V style init stuff.

touch /var/lock/subsys/local
ypdomainname job
/usr/sbin/ypserv
[root@cgsp ~]#
```

图 5-23　开机后即设定 NIS 域名

❷ /etc/ypserv.conf 文件是 NIS 服务器的主要设定文件，大部分选项都采用默认，其他稍作改动即可，配置如图 5-24 所示。解释如下:

```
dns: no
```

NIS 服务只在局域网范围内使用，因此不需要 DNS 服务。

```
files: 30
```

NIS 服务器存储的账号档案的个数，30 个账号已经足够。

```
slp: no
slp_timeout: 3600
```

设定 SLP 有关的选项，通常设为 no，即不启动 SLP 服务。

```
xft_check_port: yes
```

设定主服务器与从服务器之间同步更新账号信息库所使用的端口，此端口通常设置在小于 1024 号的端口范围内。

```
127.0.0.0/255.255.255.0 :*: *: none
192.168.10.0/255.255.255.0 : *: *: none
* :*: *: deny
```

这项是限定从服务器端和客户端的网络范围，从而建立可信任范围。第一句说明开放内部网卡，none 表示没有限制；第二句说明对局域网内所有主机都开放；第三句说明对其他任何外部 IP 地址都拒绝连接。

```
[root@cgsp etc]# vi /etc/ypserv.conf
#
# ypserv.conf   In this file you can set certain options for the NIS server,
#               and you can deny or restrict access to certain maps based
#               on the originating host.
#
#               See ypserv.conf(5) for a description of the syntax.
#

# Some options for ypserv. This things are all not needed, if
# you have a Linux net.

#               on the originating host.
#
#               See ypserv.conf(5) for a description of the syntax.
#

# Some options for ypserv. This things are all not needed, if
# you have a Linux net.

# Should we do DNS lookups for hosts not found in the hosts table ?
# This option is ignored in the moment.
dns: no

# How many map file handles should be cached ?
files: 30

# Should we register ypserv with SLP ?
slp: no
# After how many seconds we should re-register ypserv with SLP ?
slp_timeout: 3600

# xfr requests are only allowed from ports < 1024
xfr_check_port: yes

127.0.0.0/255.255.255.0 : *: *: none
192.168.10.0/255.255.255.0 : *: *: none
* : *: *: deny
```

图 5-24　/etc/ypserv.conf 的设置

❸ 设定好/etc/ypserv.conf 文件后，就可以启动 NIS 主服务器端的服务了，包括三个服务：portmap、ypserv 和 yppasswdd，如图 5-25 所示。portmap 在 NFS 部分就曾讲述过，它是负

责 RPC 的服务；ypserv 是 NIS 服务器端的主要服务；yppasswdd 提供 NIS 客户端使用者的密码修改服务，启动这个服务后，NIS 客户端可以修改 NIS 服务器端的密码。

```
[root@cgsp yp]# /etc/init.d/portmap start
Starting portmap:                                          [  OK  ]
[root@cgsp yp]# /etc/init.d/ypserv start
Starting YP server services:
[root@cgsp yp]# /etc/init.d/yppasswdd start
Starting YP passwd service:                                [  OK  ]
[root@cgsp yp]#
```

图 5-25　启动 NIS 主服务器端服务

如果要检查 NIS 的进程是否正常启动，使用下面两条命令：

```
rpcinfo -p localhost | grep yp
rpcinfo -u localhost ypserv
```

第一条命令是查看本机与 yp 相关的 rpc 进程，结果如图 5-26 所示，可以看到 yppasswdd、ypserv 和 ypbind 进程都已经启动。

```
[root@cgsp ~]# rpcinfo -p localhost |grep yp
 600100069    1   udp    984  fypxfrd
 600100069    1   tcp    986  fypxfrd
    100009    1   udp    747  yppasswdd
    100004    2   udp    935  ypserv
    100004    1   udp    935  ypserv
    100004    2   tcp    938  ypserv
    100004    1   tcp    938  ypserv
    100007    2   udp    679  ypbind
    100007    1   udp    679  ypbind
    100007    2   tcp    682  ypbind
    100007    1   tcp    682  ypbind
[root@cgsp ~]#
```

图 5-26　检查 NIS 是否正常启动

第二条命令是查看 ypserv 进程是否侦听，结果如图 5-27 所示。只有看到与图 5-26 和图 5-27 两幅图相同的结果，才表示 NIS 服务器确实是正常启动了。

```
[root@cgsp ~]# rpcinfo -u localhost ypserv
program 100004 version 1 ready and waiting
program 100004 version 2 ready and waiting
[root@cgsp ~]#
```

图 5-27　查看 ypserv 进程是否开始侦听

❹ 成功启动 NIS 相关服务后，进入/var/yp 目录，执行 make 命令刷新 NIS 数据库信息，如图 5-28 所示，表示 passwd、group 和 hosts 以及 netid 等信息都已更新，使用到的命令如下：

```
cd /var/yp
```

```
make
```

```
[root@cgsp yp]# make
gmake[1]: Entering directory `/var/yp/job'
Updating passwd.byname...
Updating passwd.byuid...
Updating group.byname...
Updating group.bygid...
Updating hosts.byname...
Updating hosts.byaddr...
Updating netid.byname...
gmake[1]: Leaving directory `/var/yp/job'
```

图 5-28　更新 NIS 数据库

点评与拓展：每次对/etc/ypserv.conf 以及其他配置文件修改后，都需要执行 make 命令使设置生效。

5.5.2　NIS 从服务器端的配置

NIS 的从服务器作为主服务器的备份服务器，提供与主服务器一样响应用户账号查询的功能。在一个比较大范围的局域网中，通常需要架设几个从服务器以防止主服务器单点失效，并分担一部分用户账号查询的负载，以避免主服务器负载过重。本节介绍如何配置 NIS 从服务器，如果读者所要配置的 NIS 服务无需配置从服务器，则可以跳过此节的阅读。

1. 主服务器端/var/yp/Makefile 文件中的 NOPUSH 项设定了主从服务器之间传送资料的方式，如果将 NOPUSH 设为 false，则主服务器允许将 NIS 数据库推送到从服务器上，这种方式较好地维持了主从服务器的数据同步，因此通常将 NOPUSH 设为 false，如图 5-29 所示。

```
[root@cgsp yp]# vi Makefile
#
# Makefile for the NIS databases
#
# This Makefile should only be run on the NIS master server of a domain.
# All updated maps will be pushed to all NIS slave servers listed in the
# /var/yp/ypservers file. Please make sure that the hostnames of all
# NIS servers in your domain are listed in /var/yp/ypservers.
#
# This Makefile can be modified to support more NIS maps if desired.
#

# Set the following variable to "-b" to have NIS servers use the domain
# name resolver for hosts not in the current domain. This is only needed,
# if you have SunOS slave YP server, which gets here maps from this
# server. The NYS YP server will ignore the YP_INTERDOMAIN key.
#B=-b
B=

# If we have only one server, we don't have to push the maps to the
# slave servers (NOPUSH=true). If you have slave servers, change this
# to "NOPUSH=false" and put all hostnames of your slave servers in the file
# /var/yp/ypservers.
NOPUSH=false
```

图 5-29　主服务器的 Makefile 设定

提及推送方式(PUSH)，与之相对应的是拖拽方式(PULL)，它们是互联网领域的两个重要的概念。拖拽方式指客户端主动向服务器端发送请求以获取所需要的信息，推送方式是指服

务器端主动将信息发送到客户端，而无需由客户端发送请求。例如，用户需要及时了解天气情况信息，他可以登录到气象台的网站去查获，这就是以拖拽方式的获得天气情况信息，也可以由气象台的网站主动发 E-mail 或以其他方式告知用户，这就是推送方式使得用户获得天气情况信息。传统互联网以拖拽方式为主，新一代互联网则有向推送方式传送信息发展的趋势。

❷ 那么主服务器是怎么规定局域网内有哪些从服务器的呢？/var/yp/ypservers 文件设定了所有的 NIS 服务器，在 ypservers 中加上图 5-30 中划线的一行，说明主机名为 node2ib.job 的这台主机是 NIS 的从服务器。

```
[root@cgsp yp]# pwd
/var/yp
[root@cgsp yp]# vi ypservers
node1ib.job
node2ib.job
~
~
```

图 5-30　主服务器 ypservers 的设定

❸ 从服务器本质上是 NIS 服务器端，因此从服务器端的设定与主服务器端的设置类似。首先也是设定 NIS 域名，这个域名必须与主服务器端一致，这里为 job，如图 5-31 所示。首先 ssh 远程登录 node2ib 这台主机，将它的 NIS 域名设为 job，并且同样在/etc/rc.local 中添加 domainname job，令其系统重启就会将 NIS 域名设为 job。

```
[root@cgsp etc]# ssh node2ib
Last login: Sat Oct 13 01:56:26 2007 from node1ib.job
-bash: /jobmgr/conf/profile.lsf: No such file or directory
[root@node2ib ~]# nisdomainname
job
[root@node2ib ~]# more /etc/rc.local
#!/bin/sh
#
# This script will be executed *after* all the other init scripts.
# You can put your own initialization stuff in here if you don't
# want to do the full Sys V style init stuff.

touch /var/lock/subsys/local
domainname job
/sbin/ypbind

route add default gw 192.168.10.1
[root@node2ib ~]#
```

图 5-31　从服务器端 NIS 域名的设定

❹ 接着在 node2ib 这台主机上，依照 5.5.1 节“NIS 主服务器端的配置”中的步骤②设定/etc/ypserv.conf 文件，该文件与主服务器设定不同的是，需要设定 trusted_master 设定项，该项指定从哪台信任的主 NIS 服务器获取数据库信息。本例设为 node1ib 即上节所设置的主 NIS 服务器的主机名。从服务器端启动与主服务器端不同，它只要启动 portmap 和 ypserv 服务，原因在于从服务器的 NIS 数据库是由主服务器端推送过来的，因此不提供 NIS 客户端修改密码的功能，所以无需启动 yppasswdd 服务。从服务器的其他配置和启动步骤在此不再赘述，读者可参照 5.5.1 节“NIS 主服务器端的配置”完成。

```
# Should we register ypserv with SLP ?
slp: no
# After how many seconds we should re-register ypserv with SLP ?
slp_timeout: 3600

# xfr requests are only allowed from ports < 1024
trusted_master: node1ib.job
xfr_check_port: yes
```

图 5-32 从服务器 ypservers 的设定

点评与拓展：从服务器本质上还是服务器端，因此配置这类从服务器通常包含两步，首先是主服务器开放从服务器功能，其次是从服务器进行类似于主服务器的配置。

5.5.3 NIS 客户端的配置

当有用户登录 NIS 客户端主机而本地查询不到用户信息时，NIS 客户端就自动连接 NFS 服务器，查询是否有该用户。连接 NFS 服务器的进程叫做 ypbind 进程，NIS 客户端的设定主要就是 ypbind 的设定；另外 NIS 客户端还提供负责查询功能的套件，叫做 yp-tools。下面介绍如何配置 NIS 客户端。

1. 首先当然是设置 NIS 的域名，与服务器端的 NIS 域名要一致，按照同样的步骤，先使用 ypdomainname 命令设定域名，然后写入/etc/rc.local 文件，让系统启动后会自动运行这个命令，如图 5-33 所示。

```
[root@cgsp ~]# ssh node3ib
Last login: Thu Oct 11 21:52:56 2007 from node1ib.job
-bash: /jobmgr/conf/profile.lsf: No such file or directory
[root@node3ib ~]# ypdomainname
job
[root@node3ib ~]# more /etc/rc.local
#!/bin/sh
#
# This script will be executed *after* all the other init scripts.
# You can put your own initialization stuff in here if you don't
# want to do the full Sys V style init stuff.

touch /var/lock/subsys/local
domainname job
/sbin/ypbind
```

图 5-33 设定客户端的 NIS 域名

另外，客户端的/etc/hosts 文件中要有 NIS 域内其他主机 IP 与主机名的对应项，如图 5-34 标注部分，就是 NIS 服务器的 IP 与主机名的对应项。

```
[root@node3ib ~]# vi /etc/hosts
 Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1        localhost.localdomain localhost
192.168.10.1     node1ib.job      node1ib      node1ib.local
192.168.10.2     node2ib.job      node2ib      node2ib.local
192.168.10.3     node3ib.job      node3ib      node3ib.local
```

图 5-34 客户端的/etc/hosts 文件

❷ 接着需要对/etc/nsswitch.conf 文件进行配置，该文件规定了 Linux 系统使用各种服务时对应的数据库及这些数据库的搜索顺序。这个设定文件功能很强大，也很复杂，不过我们只需修改 NIS 客户端经常查询的 passwd、shadow、group 和 hosts 这四个服务就可以了。在这四个服务后我们需要添加上 nis 这项。为了提高查询效率，通常将 nis 数据库写在这四个服务的第一位，如图 5-35 所示。

```
[root@node3ib ~]# vi /etc/nsswitch.conf
#
# /etc/nsswitch.conf
#
# An example Name Service Switch config file. This file should be
# sorted with the most-used services at the beginning.
#
# The entry '[NOTFOUND=return]' means that the search for an
# entry should stop if the search in the previous entry turned
# up nothing. Note that if the search failed due to some other reas
# (like no NIS server responding) then the search continues with th
# next entry.
#
passwd:     nis db files nisplus
shadow:     nis db files nisplus
group:      nis db files nisplus

#hosts:      db files nisplus nis dns
hosts:      nis db files nisplus dns
#hosts:       files dns
```

图 5-35　/etc/nsswitch.conf 文件设定

❸ 接下来是修改/etc/sysconfig/authconfig 文件，只要将这个文件的 USENIS 选项设定为 yes 即可，如图 5-36 所示。

```
[root@node3ib ~]# vi /etc/sysconfig/authconfig
USEMD5=yes
USECRACKLIB=yes
USEDB=no
USEHESIOD=no
USELDAP=no
USENIS=yes
USEPASSWDQC=no
USEWINBIND=no
USEKERBEROS=no
USELDAPAUTH=no
USESHADOW=yes
USESMBAUTH=no
USEWINBINDAUTH=no
```

图 5-36　/etc/sysconfig/authconfig 文件的设定

❹ 然后需要在/etc/pam.d/system-auth 文件中添加 nis 一项，这里在 PAM 设定栏后添加 NIS 服务，找到图 5-37 中标注的那一行，在这行的最后添加 nis 即可。

❺ 以上几个配置文件使系统在各方面对 NIS 进行了支持，但是 NIS 客户端的主要设定文件是/etc/yp.conf，需要在这个文件里正确设定域名和域内的 NIS 服务器，添加如图 5-38 标注的两句话。第二句 domain NISDOMAIN broadcast 的意思是是否采用广播的方式查询 NIS 数据库，当域内有多台 NIS 服务器即采用主从服务器机制时，就需要采用广播方式，这样所有的 NIS 服务器才都会收到客户端的查询请求。

```
[root@node3ib ~]# vi /etc/pam.d/system-auth
#%PAM-1.0
# This file is auto-generated.
# User changes will be destroyed the next time authconfig is run.
auth        required      /lib/security/$ISA/pam_env.so
auth        sufficient    /lib/security/$ISA/pam_unix.so likeauth nullok
auth        required      /lib/security/$ISA/pam_deny.so

account     required      /lib/security/$ISA/pam_unix.so
account     sufficient    /lib/security/$ISA/pam_succeed_if.so uid < 100 quiet
account     required      /lib/security/$ISA/pam_permit.so

password    requisite     /lib/security/$ISA/pam_cracklib.so retry=3
password    sufficient    /lib/security/$ISA/pam_unix.so nullok use_authtok md5 shadow nis
password    required      /lib/security/$ISA/pam_deny.so

session     required      /lib/security/$ISA/pam_limits.so
session     required      /lib/security/$ISA/pam_unix.so
```

图 5-37　/etc/pam.d/system-auth 文件的设定

❻ 通过以上几步的配置工作，就可以启动 ypbind 服务了，ypbind 服务也是基于 RPC 机制的，因此需要先启动 portmap 服务，如图 5-39 所示。

📖**点评与拓展**：NIS 的客户端设定看起来比较复杂，但是系统设定的三个文件：nsswitch.conf、authconfig 和 system-auth，都只需要添加上 NIS 服务相关项，实际上是非常简单的。

```
[root@node3ib ~]# vi /etc/yp.conf
# /etc/yp.conf - ypbind configuration file
# Valid entries are
#
 domain job server node1ib
#       Use server HOSTNAME for the domain NISDOMAIN.
#
# domain NISDOMAIN broadcast
#       Use  broadcast  on  the local net for domain NISDOMAIN
#
# domain NISDOMAIN slp
#       Query local SLP server for ypserver supporting NISDOMAIN
#
#       Use server HOSTNAME for the  local  domain.  The
#       IP-address of server must be listed in /etc/hosts.
#
# broadcast
#       If no server for the default domain is specified or
#       none of them is rechable, try a broadcast call to
#       find a server.
#
ypserver node1ib
```

图 5-38　/etc/yp.conf 的设定

```
[root@node3ib ~]# /etc/init.d/portmap start
Starting portmap:                                          [  OK  ]
[root@node3ib ~]# /etc/init.d/ypbind start
Binding to the NIS domain:
Listening for an NIS domain server.
[root@node3ib ~]#
```

图 5-39　启动 ypbind 服务

5.5.4　NIS 服务客户端的检验

NIS 服务还提供了客户端的检验工具，这些工具包括 yptest、ypwhich 和 ypcat，本节将分别对其作简单的介绍。

yptest 检验 NIS 设定的所有参数，部分测试结果如图 5-40 所示。测试的第 3 步，通常

会出现在 passwd.byname 中找不到 nobody 字样的情况，这是因为早期 Linux 系统将 nobody 用户的 UID 设为了 65534，而 Red Hat 和 Fedora 6.0 则将 nobody 用户的 UID 设为了 99，因此才出现了这样的警告，不过这个警告不会影响 NIS 服务。最关键的测试在第 9 步，它测试了所有的 NIS 账号，如果第 9 步能够列出 NIS 服务器上的所有账户，表示检验成功，如图 5-41 所示。

```
[root@node3ib ~]# yptest
Test 1: domainname
Configured domainname is "job"

Test 2: ypbind
Used NIS server: node1ib.job

Test 3: yp_match
WARNING: No such key in map (Map passwd.byname, key nobody)

Test 4: yp_first
mysql mysql:!!:502:502::/home/mysql:/bin/bash
```

图 5-40 yptest 命令

```
Test 9: yp_all
mysql mysql:!!:502:502::/home/mysql:/bin/bash
nistest nistest:$1$nPyoTpuZ$yZ4TLGCcy05j3iuDzmAZV1:503:503::/home/users/nistest:/bin/bash
test test:$1$WtCxBCOS$0EjpWi2D9qaOpTktjsERl.:500:500::/home/test:/bin/bash
globus globus:!!:501:501::/home/globus:/bin/bash
```

图 5-41 yptest 第 9 步的测试结果

ypwhich 命令提供了对 NIS 服务器数据库名称和数量的检验功能，带上-x 参数显示 NIS 客户端需要与服务器端交互的数据库有哪些，如图 5-42 所示。

```
[root@node3ib ~]# ypwhich -x
Use "ethers"    for map "ethers.byname"
Use "aliases"   for map "mail.aliases"
Use "services"  for map "services.byname"
Use "protocols" for map "protocols.bynumber"
Use "hosts"     for map "hosts.byname"
Use "networks"  for map "networks.byaddr"
Use "group"     for map "group.byname"
Use "passwd"    for map "passwd.byname"
[root@node3ib ~]#
```

图 5-42 ypwhich 命令

ypcat 命令提供读取上述列出的 NIS 数据库的内容，命令格式如下：

```
ypcat [-h NIS 服务器名称] [数据库名称]
```

图 5-43 显示了使用 ypcat 命令读取到本地 passwd.byname 和 hosts.byname 两个数据库的内容，可以看到保存密码的 passwd.byname 内容是用户账号名和加密的密码，而 hosts.byname 中存放了 NIS 服务器的 IP 和主机名。

```
[root@node3ib ~]# ypcat passwd.byname
mysql:!!:502:502::/home/mysql:/bin/bash
nistest:$1$nPyoTpuZ$yZ4TLGCcy05j3iuDzmAZV1:503:503::/home/users/nistest:/bin/bash
test:$1$WtCxBCOS$0EjpWi2D9qaOpTktjsERl.:500:500::/home/test:/bin/bash
globus:!!:501:501::/home/globus:/bin/bash
[root@node3ib ~]# ypcat hosts.byname
192.168.10.8     node8ib.job      node8ib node8ib.local
```

图 5-43　ypcat 命令

点评与拓展： Linux 系统提供的这三条命令可以全面检验 NIS 服务器端是否正常工作，yptest 提供的是设定参数的检验，ypwhich 用于检验 NIS 数据库，ypcat 可以具体查看数据库的内容。

5.6　结合 NFS 和 NIS 管理系统用户

应用实例导航——统一管理主从架构网络中的账户

※场景呈现

让我们回到某机构所购置的高性能计算设备上，该机构发现 32 个紧耦合的从计算结点不仅需要具备同样的文件系统，也需要具有相同的系统用户。也就是说，在主控结点能够成功登录的用户，能够直接登录到所有的从计算结点上去，而不需要通过额外的创建用户操作。网络拓扑依然如图 5-14 所示。

请网络管理员利用结合 NFS 和 NIS 方面的知识，实现系统用户的统一管理，而且每个系统用户在所有结点具有同样的根目录。

※技术要领

(1) 利用 NFS 挂载系统用户根目录。
(2) 利用 NIS 统一管理系统用户。

配置和启动好服务器和客户端的 NIS 服务后，本节结合 NFS 和 NIS 服务统一管理用户账号，以说明 NFS 和 NIS 在大规模网络服务器管理上的优势。在 5.3.1 节“主从架构下 NFS 服务的需求”所介绍的主从架构的拓扑结构中，主结点对从结点具有控制权，其中包括对用户账户的控制，也就是说在主结点上所创建的用户能够以同样的目录、同样的权限在从结点上使用，NFS 和 NIS 的结合为这种账户管理方式提供了一种简明的解决方案，那就是利用 NFS 挂载统一的用户目录，利用 NIS 统一地创建用户数据库。下面介绍这种方案的配置及其效果。

❶ 5.2.1 节“NFS 服务器端的配置”中所配置的/etc/exports 文件中有一行为：

```
/home/users 192.168.10.0/255.255.255.0 (rw,insecure,no_root_squash,asyn c)
```

它表示将/home/users 目录设为局域网内可读可写的共享目录，而这个目录就是为 NIS 准备的，这里面存放了所有 NIS 服务器上用来创建用户的根目录。

❷ 首先将/home/users 目录挂载到从结点上，如图 5-44 中第一条横线的标注部分所示，所使用的挂载命令已在 5.2.2 节“NFS 客户端的配置和测试”中介绍过。

```
[root@cgsp admin]# mkdir /home/users
[root@cgsp admin]# sshrrunon "mount -t nfs 192.168.10.1:/home/users /home/users" 2 3
---------------sshrun mount -t nfs 192.168.10.1:/home/users /home/users on node2ib.job------
---------------sshrun mount -t nfs 192.168.10.1:/home/users /home/users on node3ib.job------
[root@cgsp admin]# useradd nistest -d /home/users/nistest
[root@cgsp admin]# ls /home/users
nistest
[root@cgsp admin]# passwd nistest
Changing password for user nistest.
New UNIX password:
BAD PASSWORD: it is based on a dictionary word
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
```

图 5-44　挂载/home/users 目录并创建新用户

接着可以在主控结点上，即 NIS 服务器端，创建 nistest 用户，并修改其密码，如图 5-44 所示。两条命令解释如下:

```
useradd nistest -d /home/users/nistest
```

useradd 或 adduser 命令是系统创建用户的命令，-d 参数后附带了该用户的根目录，如果不带-d 参数，用户默认的根目录将是/home/nistest。这里我们需要将其放到共享目录/home/users 下面。

请注意，以后所有创建的系统用户不能默认将其根目录设在/home 下，这样将导致从结点无法访问它；必须通过-d 参数将根目录设在/home/users 下，以便所有结点都能通过 NFS 共享目录访问。

```
passwd nistest
```

这条命令是为 nistest 修改密码用的，只有 root 用户才有权限执行此命令。

❸ NIS 服务启动后，就可以尝试使用 nistest 这个用户登录 node3ib 这个从结点，如图 5-45 所示，正确输入密码后成功登录，并进入默认的/home/users/nistest 目录。这说明 node3ib 这个结点上本来没有的用户 nistest，可以通过 NIS 系统在服务器端被查询到，并且根目录被 NFS 成功设为/home/users/nistest 这个共享目录。

```
[root@cgsp nistest]# ssh node3ib -l nistest
nistest@node3ib's password:
-bash: /jobmgr/conf/profile.lsf: No such file or directory
[nistest@node3ib ~]$ cd
[nistest@node3ib ~]$ pwd
/home/users/nistest
[nistest@node3ib ~]$ 
```

图 5-45　使用 nistest 用户登录 node3ib 主机

❹ nistest 用户登录任何从结点后，在/home/user/nistest 这个根目录下具有管理员权限，可以创建和删除目录及文件，并且所创建的目录将被其他所有主机上的 nistest 所访问。使用 mkdir

命令在 node2ib 这个结点上创建一个名为 allknow 的目录，如图 5-46 所示。然后登录主控结点，列出/home/user/nistest 后，可以发现在主控结点依然可以看到刚才所创建的 allknow 目录，如图 5-47 所示。

```
[nistest@node2ib ~]$ pwd
/home/users/nistest
[nistest@node2ib ~]$ mkdir allknow
[nistest@node2ib ~]$ ls
allknow
[nistest@node2ib ~]$
```

图 5-46　nistest 用户在 node2ib 上创建目录

```
[root@cgsp ~]# ls /home/users/nistest/
allknow
[root@cgsp ~]#
```

图 5-47　任意台主机都能列出所创建的目录

点评与拓展：这个例子看起来很简单，但它建立在对 NFS 和 NIS 服务全面掌握的基础上，也是当今集群系统管理系统用户的普遍方案，值得读者悉心体会。

5.7　本 章 小 结

本章主要介绍了网络文件系统 NFS 和 NIS 服务的相关原理及安装配置方法。读者需要注意的是，NFS 虽是一个网络文件共享协议，但并没有提供数据传输功能，而是借助 RPC(远程过程调用)机制来实现数据传输。本章还特别介绍了主从架构下的 NFS 服务，这是一般同类参考书中所没有的，主要是考虑到主从结构在实际应用中越来越广泛，尤其是集群(cluster)系统。NIS 服务主要应用于具有多台 Linux 服务器的大型网络，对所有 Linux 主机的用户账号数据进行集中统一管理，提高了资源利用率，也相应节省了 Linux 主机的硬盘空间。最后，读者还要充分认识到结合 NFS 和 NIS 对系统用户进行管理的优越性。

第 6 章　DHCP 服务器的配置与架设

DHCP(Dynamic Host Configuration Protocol)，即动态主机配置协议，主要用于简化主机 IP 地址的分配管理。DHCP 服务器可以自动将网络参数(如 IP 地址、子网掩码、默认网关、DNS 地址等)正确地分配给网络中的每台计算机，使客户端计算机在开机时就自动设定好相关网络参数值。本章主要介绍了 DHCP 服务中涉及的重要概念及其基本工作原理。重点介绍了 DHCP 服务器端动态 IP 地址分配和静态 IP 地址分配的配置方法，最后介绍 Linux 和 Windows 两种不同操作系统的 DHCP 客户端配置。

通过本章的学习，读者应掌握以下内容：

- DHCP 的工作原理
- DHCP 服务器端动态 IP 地址分配的配置
- DHCP 服务器端静态 IP 地址分配的配置
- DHCP 客户端的配置

6.1　DHCP 服务概述

6.1.1　DHCP 服务简介

DHCP 采用 Client/Server 模式，安装了 DHCP 服务软件的主机就是 DHCP 服务器，而启用了 DHCP 功能的主机就是 DHCP 客户端。客户端启动时，自动和 DHCP 服务器端通信，DHCP 服务器则为客户端自动分配 IP 地址。

DHCP 服务器以地址租约的方式来为 DHCP 客户端提供服务。DHCP 服务器分配给客户端的 IP 类型主要有以下两种。

- 固定 IP(static IP)：一般来说，只要客户端计算机的网卡不换，那么它的 MAC(传输媒体访问控制)地址就不会改变。DHCP 服务器可以根据 MAC 地址来分配固定的 IP 地址，因此该客户机就能每次都以一个固定的 IP 连接上 Internet。在 Linux 上可以使用 ifconfig 及 arp 来获取本机 MAC 地址。
- 动态 IP (dynamic IP)：客户端每次连上 DHCP 服务器所取得的 IP 地址都不是固定的，而是由 DHCP 服务器从尚未被使用的 IP 地址中随机选取。

需要注意的是，除非局域网内的某台计算机有可能用作一些网络服务的主机，否则使用动态 IP 的设定比较简单，而且使用上具有较好弹性。

DHCP 的优点是“免客户端设定”，非常方便移动上网。架设 DHCP 服务器的最佳场合

是使用较多的移动设备的场合，例如，某公司内部有很多使用笔记本电脑的地方(笔记本电脑是典型的移动设备)，如果每到一处都要重新设定网络参数，还要担心是否会与别人的 IP 地址相冲突，显然十分麻烦，这时候 DHCP 是最好的解决办法。或者是网络上计算机数量相当多的场合，当一个网络内计算机数量相当庞大时，难以一个一个来设定它们的网络参数，这时候为节省时间和减少麻烦，架设 DHCP 是非常方便的措施。

6.1.2　DHCP 工作原理

DHCP 通常是局域网内的一个通信协议。DHCP 服务器与客户端在同一个局域网内，先由客户端向整个局域网内的所有主机广播 DHCP 请求信息，当网络中有 DHCP 主机时，才会响应客户端的 IP 参数请求。如图 6-1 所示，客户端取得 IP 参数的过程如下。

❶ 客户端发送 DHCP 请求

如果客户端设定使用 DHCP 取得 IP，则当客户端开机或者是重启网卡时，客户端主机会发送 DHCP 请求给局域网内的所有计算机。DHCP 请求信息的目标 IP 是 255.255.255.255，普通主机接收到这个请求后会直接予以丢弃。

❷ DHCP 主机响应请求

DHCP 主机在接收到客户端的请求后，会根据客户端的硬件地址(MAC)与本身的设定数据来进行下列工作：①到 DHCP 服务器的登录文件中查询该用户是否曾经用过某个 IP，如果是且该 IP 目前无人使用，则提供该 IP 给客户端；②如果服务器已经设定对该 MAC 地址提供固定 IP 时，则提供设定的固定 IP；③如果不符合前述任一条件，则随机选取目前没有被使用的 IP 给用户，并记录下来。

此外，DHCP 服务器还会提供一个租约时间给客户端，并等待客户端的响应。

❸ 客户端接受本次取得的 IP 并开始设定本身的网络环境，包括改写 /etc/resolv.conf 等；并且会向 DHCP 服务器发送一个确认信息，确认该参数已被接受。

❹ DHCP 服务器记录该次租约行为，回送应答确认

客户端发送确认信息建立租约行为后，该次租约会被记录到服务器的登录文件上，并且开始租约计时。同时，会以广播方式发送一个应答信息给 DHCP 客户端，客户端收到应答信息后，就完成了获得 IP 地址的整个过程。



图 6-1　DHCP 工作流程

以上就是 DHCP 协议在 Server 端与 Client 端的工作流程。从上述工作流程看，只要 Server 端设定没有问题，且 Server 与 Client 在硬件联机上也没问题，那么 Client 就可以直接通过 Server 来取得网络参数。

6.2 DHCP 服务器端的配置

应用实例导航——A 公司 DHCP 服务器的架设

※场景呈现

A 公司需要在内部局域网架设 DHCP 服务器，局域网段为 172.18.12.1～172.18.12.255。DHCP 服务器将动态分配的 IP 地址范围限制在 172.18.12.220～172.18.12.240 之内，其他 IP 地址保留下来；路由设为 172.18.12.129；DNS 服务器设为 202.119.24.12。网络拓扑图如图 6-2 所示。

另外，DHCP 分配的 IP 的预设租约期是 10 万秒，最长为 20 万秒。

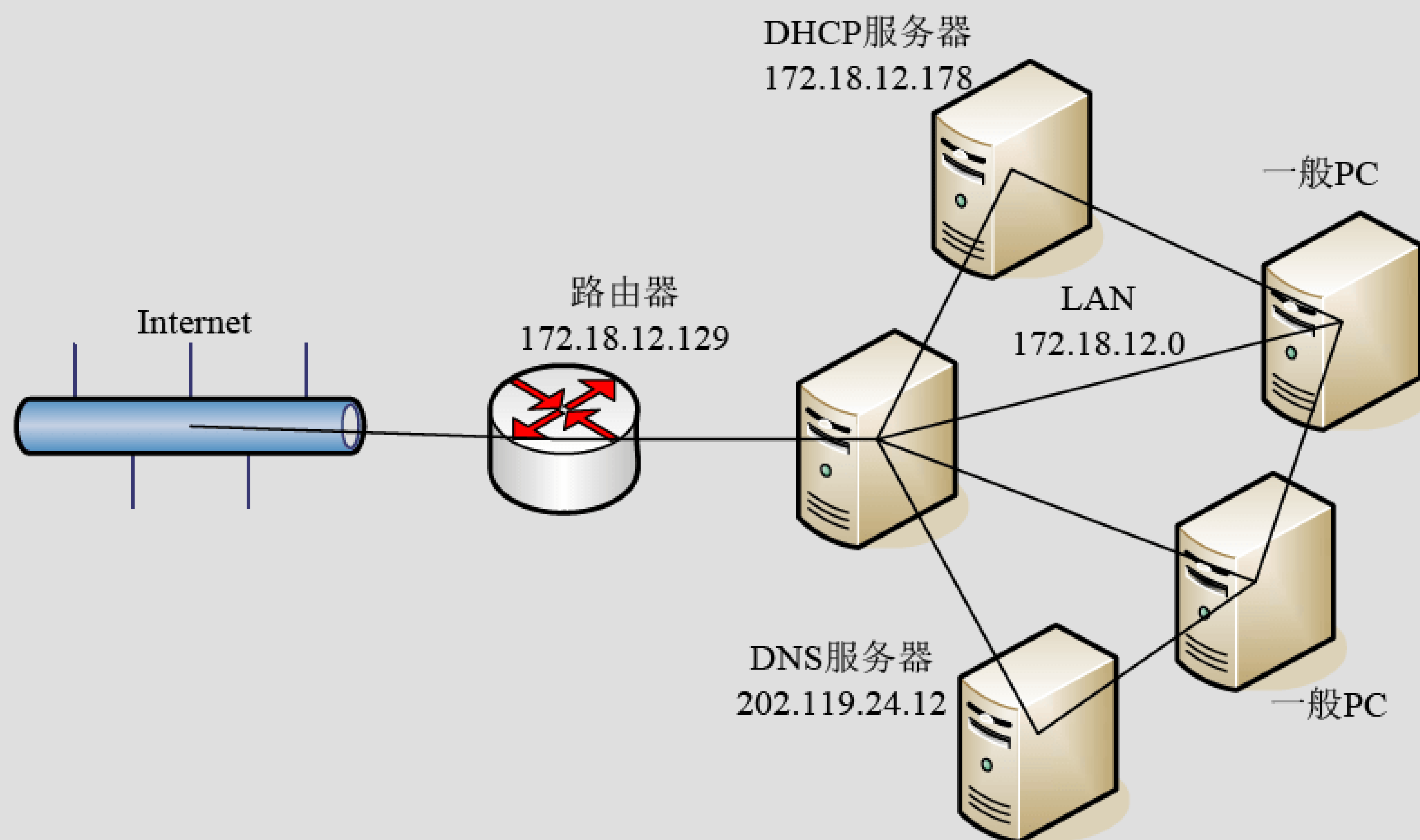


图 6-2　DHCP 服务器拓扑图

※技术要领

(1) DHCP 服务器的安装。
(2) DHCP 服务器端的配置。
(3) DHCP 服务器端的启动和测试。

DHCP 服务器为局域网内的主机动态分配 IP，使用户无需手动配置，方便了局域网的

管理和用户的使用。本节介绍 DHCP 服务器端的配置方法，首先介绍 Linux 系统下 DHCP 软件所包括的一些组件，使读者对 DHCP 有整体上的了解；然后介绍 DHCP 服务器的配置，以满足场景显现中的要求；最后介绍如何启动 DHCP 服务器以及如何测试 DHCP 服务器是否成功启动。

6.2.1　DHCP 软件结构简介

DHCP 软件方面的要求很简单，只需服务器端软件 dhcp。dhcp 是 DHCP 服务器的主要设定文件、启动脚本和执行文件，而且我们只需要这一个软件即可。安装方式上可以直接使用原版光盘 rpm 来安装。整个软件的结构分以下三个部分。

- /etc/dhcpd.conf
  这是 dhcp 服务器的主要设定文件。在某些 Linux 版本上这个文件可能不存在，因此，如果已经安装了 dhcp 软件却找不到这个文件，手动自行建立即可。该文件的设定很简单，设定文件的实际位置时需要参考 /etc/init.d/dhcpd 的规范。
- /usr/sbin/dhcpd
  这是启动整个 dhcp daemon 的执行文件。最详细的执行方式可以查阅 man dhcpd。
- /var/lib/dhcp/dhcpd.leases

该文件用于记录 DHCP 服务器端与客户端租约建立的起讫日期，租约期限由 DHCP 服务器所指定，规定了客户端从 DHCP 服务器获得的 IP 地址的可使用时间长度。如果客户端在租约到期前，尚未更新租约，则 DHCP 服务器将会收回分配给该客户端的 IP 地址。

6.2.2　DHCP 服务器端的配置

有了对 DHCP 软件的大概了解后，我们就开始介绍 DHCP 服务器的安装和配置，具体根据下面的步骤。

1. Fedora 6.0 没有默认安装 DHCP 服务，需要手动安装；在 Fedora 6.0 的安装光盘或者本书附带光盘中获得 DHCP 的 rpm 安装包，名为 dhcp-3.0.4-21.fc6.i386.rpm。rpm 包的通常安装方法是输入 rpm-i name.rpm 进行安装，但是对于 DHCP 的安装包，这种安装方式将报错，错误类型为 warning: dhcp-3.0.4-21.fc6.src.rpm: Header V3 DSA signature: NOKEY, key ID 4f2a6fd2，如图 6-3 所示。这种错误也是安装 rpm 包时经常会碰到的错误之一，这时就要使用下面命令安装，如图 6-4 所示。

```
rpm -ivh dhcp-3.0.4-21.fc6.i386.rpm -force -nodeps
```

```
[root@seugrid3 local]# rpm -ivh dhcp-3.0.4-21.fc6.src.rpm
warning: dhcp-3.0.4-21.fc6.src.rpm: Header V3 DSA signature: NOKEY, key ID 4f2a6fd2
   1:dhcp                  warning: user brewbuilder does not exist - using root
```

图 6-3　安装 DHCP 出错

可以使用下面命令查看 DHCP 服务是否安装成功:

```
rpm -qa | grep dhcp
```

结果如图 6-5 所示，dhcp-3.0.4-21.fc6 就是刚才所安装的 DHCP 服务。

```
[root@seugrid3 local]# rpm -ivh dhcp-3.0.4-21.fc6.i386.rpm --force --nodeps
warning: dhcp-3.0.4-21.fc6.i386.rpm: Header V3 DSA signature: NOKEY, key ID 4f2a6fd2
Preparing...                ########################################### [100%]
   1:dhcp                   ########################################### [100%]
```

图 6-4　正确安装 DHCP

```
[root@seugrid3 local]# rpm -qa | grep dhcp
dhcpv6_client-0.10-16.1
dhcp-3.0.4-21.fc6
[root@seugrid3 local]#
```

图 6-5　检查 DHCP 是否成功安装

❷ 从 DHCP 软件结构简介中知道，它的最主要的配置文件为/etc/dhcpd.conf，这个文件没有默认设置，需要手动创建，添加如图 6-6 所示的内容。该配置文件“#”号开头的为注释行，每行以分号结束。

```
[root@seugrid2 ~]# vi /etc/dhcpd.conf
#
# DHCP Server Configuration file.
#   see /usr/share/doc/dhcp*/dhcpd.conf.sample
#
ddns-update-style none;
default-lease-time 100000;
max-lease-time     200000;
option routers     172.18.12.129;
option broadcast-address 172.18.12.255;
option domain-name-servers 202.119.24.12;


subnet 172.18.12.0 netmask 255.255.255.0 {
   range 172.18.12.200 172.18.12.240;
   option subnet-mask 255.255.255.128;
   option nis-domain "seugrid2.seu.edu.cn";
   option domain-name "seu.edu.cn";
}
```

图 6-6　/etc/dhcpd.conf 设置

第一段标注部分是 DHCP 的整体环境设定，逐行解释如下：

```
ddns-update-style none;
```

不需要更新 DDNS 的设定。

```
default-lease-time 100000;
```

分配 IP 的预设有效期为 10 万秒。

```
max-lease-time 200000;
```

分配 IP 的最大有效期为 20 万秒。

```
option routers   172.18.12.129;
```

路由地址为 172.18.12.129。

```
option broadcast-address 172.18.12.255;
```

设置了广播地址，以此地址发 IP 包，局域网内所有主机都能收到。

```
option domain-name-servers 202.119.24.12;
```

设置了 DNS 服务器的 IP 地址，如果有多个 DNS 服务器地址，则多个 IP 地址间用“,”隔开。

第二段标注部分是设定动态分配 IP 的，详细解释如下：

```
subnet 172.18.12.0 netmask 255.255.255.0 {
  range 172.18.12.200 172.18.12.240;
```

这两行的设置至关重要，必须两行匹配。第一行设定了子网号和子网掩码，第二行设定了动态分配 IP 的范围，其必须在第一行所设定的子网 IP 范围内，否则将导致 DHCP 服务启动不成功，比如说，如果第一行改为 subnet 172.18.12.0 netmask 255.255.255.128，由第 2 章的讲述我们知道，子网 IP 地址的前 25 位表示子网号，后面的 7 位表示主机号，因此，此时主机的最大 IP 应为 172.18.12.128，下面所设置的 172.18.12.200 这个 IP 就不在上述子网的 IP 范围之内。此时，DHCP 服务将无法启动。

```
option subnet-mask 255.255.255.128;
option nis-domain "seugrid2.seu.edu.cn";
option domain-name "seu.edu.cn";
}
```

以上三项为可选参数，非关键设置，解释从略。

❸ /etc/hosts 文件记录了 IP 和主机名的对应关系，它能提高 DNS 查询的效率。对于 IP 动态分配的情况，最好是将动态 IP 范围内的所有 IP 都写入/etc/hosts 文件，然后随意取个主机名，可以如图 6-7 所示那样设定。

```
[root@seugrid2 ~]# vi /etc/hosts
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1       localhost.localdomain   localhost
172.18.12.178   seugrid2.seu.edu.cn     seugrid2
172.18.12.181   seugrid1.seu.edu.cn     seugrid1


172.18.12.140 dynamic01
172.18.12.220 dynamic02
172.18.12.221 dynamic03
172.18.12.222 dynamic04
172.18.12.223 dynamic05
172.18.12.224 dynamic06
```

图 6-7 设定/etc/hosts 文本

📖**点评与拓展**：DHCP 的文件配置与 Linux 基础网络配置中提到的概念密切相关，如路由地址、广播地址、子网、子网掩码和 DNS 地址等，准确地理解这些基本概念，有助于正确地设定 DHCP 服务。

6.2.3 DHCP 服务器的启动和测试

设定好/etc/dhcpd.conf 之后，就可以启动 DHCP 服务器了，DHCP 的成功启动与 6.2.2 节所讲的/etc/dhcpd.conf 文件和/etc/sysconfig/dhcpd 文件有关。

❶ 对于多网卡的主机，我们需要对 DHCP 服务在哪块网卡上侦听作设定。在/etc/sysconfig/dhcpd 文件中的 DHCPDARGS 参数后面添加我们需要侦听的网卡号，本例的主机用 eth0 的网卡接入局域网，因此，将 DHCPDARGS 设为 eth0，如图 6-8 所示。实际上，dhcpd 执行脚本在第 58 行左右读取 DHCPDARGS 中的网卡号，因此也可以直接在 dhcpd 脚本上作修改，如图 6-9 所示。

```
[root@seugrid2 ~]# vi /etc/sysconfig/dhcpd
# Command line options here
DHCPDARGS="eth0"
~
~
```

图 6-8　多网卡主机的 dhcpd 服务设定

```
start() {
        # Start daemons.
        echo -n $"Starting $prog: "
        daemon /usr/sbin/dhcpd ${DHCPDARGS} 2>/dev/null
        RETVAL=$?
        echo
```

图 6-9　在 dhcpd 脚本中的 DHCPDARGS 参数

❷ 对于多网卡的主机也设定无误后，就可以用下面命令启动 dhcpd 服务了，启动和停止 dhcpd 服务的执行结果如图 6-10 所示。关闭、重启 dhcpd 服务，查看 dhcpd 服务状态等命令格式亦列于下方。

```
    /etc/init.d/dhcpd start
或者 service dhcpd start
   /etc/init.d/dhcpd  {restart|reload|condrestart|status}
```

```
[root@seugrid2 ~]# /etc/init.d/dhcpd start
Starting dhcpd:                                            [  OK  ]
[root@seugrid2 ~]# /etc/init.d/dhcpd stop
Shutting down dhcpd:                                       [  OK  ]
```

图 6-10　启动 dhcpd 服务

❸ /var/log/messages 文件记录了 DHCP 服务的日志，如果 DHCP 不能成功启动，/var/log/messages 文件中就记录了错误信息，查询它有助于我们走出泥潭。由于类似于/var/log/messages 的日志文件一般都比较长，最后几行才记录了最近一次的日志信息，因此一般不使用 more 命令查看日志文件，而是使用 tail 命令:

```
tail -n 10 /var/log/messages
```

该命令的意思是查看/var/log/messages 文件的最后 10 行，结果如图 6-11 所示。可以看到最后一行显示 dhcpd startup succeeded，表示 dhcpd 服务启动成功了。

另外，还可以通过查看网络端口来确定 dhcpd 是否启动成功。DHCP 服务在 UDP 的 67 号端口侦听，因此可以通过下面命令查看：

```
netstat -tulnp | grep 67
```

图 6-11 显示了运行结果，dhcpd 已经成功运行，进程号为 32172。

```
[root@seugrid2 ~]# tail -n 10 /var/log/messages
Oct 14 19:38:50 seugrid2 dhcpd: dhcpd shutdown succeeded
Oct 14 19:39:53 seugrid2 dhcpd: Internet Systems Consortium DHCP Server V3.0.4-RedHat
Oct 14 19:39:53 seugrid2 dhcpd: Copyright 2004-2006 Internet Systems Consortium.
Oct 14 19:39:53 seugrid2 dhcpd: All rights reserved.
Oct 14 19:39:53 seugrid2 dhcpd: For info, please visit http://www.isc.org/sw/dhcp/
Oct 14 19:39:53 seugrid2 dhcpd: Wrote 0 leases to leases file.
Oct 14 19:39:53 seugrid2 dhcpd: Listening on LPF/eth0/00:12:3f:c3:30:6b/172.18.12/24
Oct 14 19:39:53 seugrid2 dhcpd: Sending on   LPF/eth0/00:12:3f:c3:30:6b/172.18.12/24
Oct 14 19:39:53 seugrid2 dhcpd: Sending on   Socket/fallback/fallback-net
Oct 14 19:39:53 seugrid2 dhcpd: dhcpd startup succeeded
[root@seugrid2 ~]# netstat -tulnp |grep 67
udp        0      0 0.0.0.0:67                  0.0.0.0:*                           32172/dhcpd
You have new mail in /var/spool/mail/root
[root@seugrid2 ~]#
```

图 6-11　dhcpd 服务启动成功

4. 如果 6.2.2 节“DHCP 服务器端的配置”中的步骤(2)中所述将子网掩码设为 255.255.255.128 的话，启动 DHCP 服务时就会出现如图 6-12 标注部分所示的错误，意思是所要动态分配的 IP 范围不在子网范围之内。

```
[root@seugrid2 ~]# more /var/log/messages
Oct 14 04:02:48 seugrid2 syslogd 1.4.1: restart.
Oct 14 14:58:05 seugrid2 dhcpd: Internet Systems Consortium DHCP Server V3.0.4-RedHat
Oct 14 14:58:05 seugrid2 dhcpd: Copyright 2004-2006 Internet Systems Consortium.
Oct 14 14:58:05 seugrid2 dhcpd: All rights reserved.
Oct 14 14:58:05 seugrid2 dhcpd: For info, please visit http://www.isc.org/sw/dhcp/
Oct 14 14:58:05 seugrid2 dhcpd: Address range 172.18.12.200 to 172.18.12.240 not on net 172.18.12.0/255.255.255.128!
Oct 14 14:58:05 seugrid2 dhcpd:
```

图 6-12　dhcpd 启动的错误日志

点评与拓展： 如果是单网卡的主机作为 DHCP 服务器，可以忽略本节所述的第(1)步。另外，DHCP 启动不成功可以及时查询日志文件。

6.3　DHCP 客户端的配置

应用实例导航——A 公司 DHCP 客户端的配置

※场景呈现

A 公司架设好主从 DHCP 服务器后，接着就要对客户端进行配置，客户端包括 Windows、Linux 两种服务器；试分别写出配置 Windows 和 Linux 客户端的步骤。

当某台客户端连接上 DHCP 服务器获得动态 IP 地址后，可以获得该台客户端的 MAC 地址，要求配置 DHCP 服务器端，实现下次该客户端连接 DHCP 服务器时，静态为它分配 IP 172.18.12.140。

※技术要领

(1) Linux 客户端的配置。
(2) Windows 客户端的配置。
(3) DHCP 静态 IP 的配置。

6.3.1 Linux 客户端的配置

本节讲述如何在 Linux 客户端下面设置 DHCP 服务，这个步骤很简单，但是需要注意的是，这些设置都需要在本机上操作，因为一旦使用 DHCP 就与网络断开连接，远程操纵将失效。

1. Linux 客户端的设置很简单，只要设置/etc/sysconfig/network-scripts/ifcfg-eth0 文件即可，找到该文件中的 BOOTPROTO 这个参数，改成 dhcp 就设置成功了，如图 6-13 标注部分所示。

```
[root@sec local]# vi /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=dhcp
IPADDR=172.18.12.179
NETMASK=255.255.255.128
GATEWAY=172.18.12.129
TYPE=Ethernet
USERCTL=no
IPV6INIT=no
PEERDNS=yes
```

图 6-13　ifcfg-eth0 文件的设置

2. 为了使网络设置生效，使用下面命令重启网络服务，如图 6-14 所示。

```
/etc/init.d/network restart
```

```
~
[root@sec local]# /etc/init.d/network restart
```

图 6-14　重启网络服务

点评与拓展： 重启 Linux 服务器网络服务时，应在本机前操作，不要采用远程操作功能，因为网络重启时将会使 SSH 断开连接。

6.3.2　Windows 客户端的配置

在 DHCP 服务器端成功启动后，局域网的任一台主机上可以通过该 DHCP 服务器获得 IP 地址，这个主机的系统可以是 Linux 系统，也可以是 Windows XP 系统，这里介绍 Windows XP 系统的测试过程。

❶ 右击【网上邻居】图标，在弹出的快捷菜单中选择【属性】命令，弹出【网络连接】窗口，如图 6-15 所示。右击【本地连接】图标，在弹出的快捷菜单中选择【属性】命令，如图 6-15 标注部分所示，弹出【本地连接 属性】窗口，如图 6-16 所示。选中【Internet 协议（TCP/IP）】复选框，单击【属性】按钮，如图 6-16 标注部分所示。

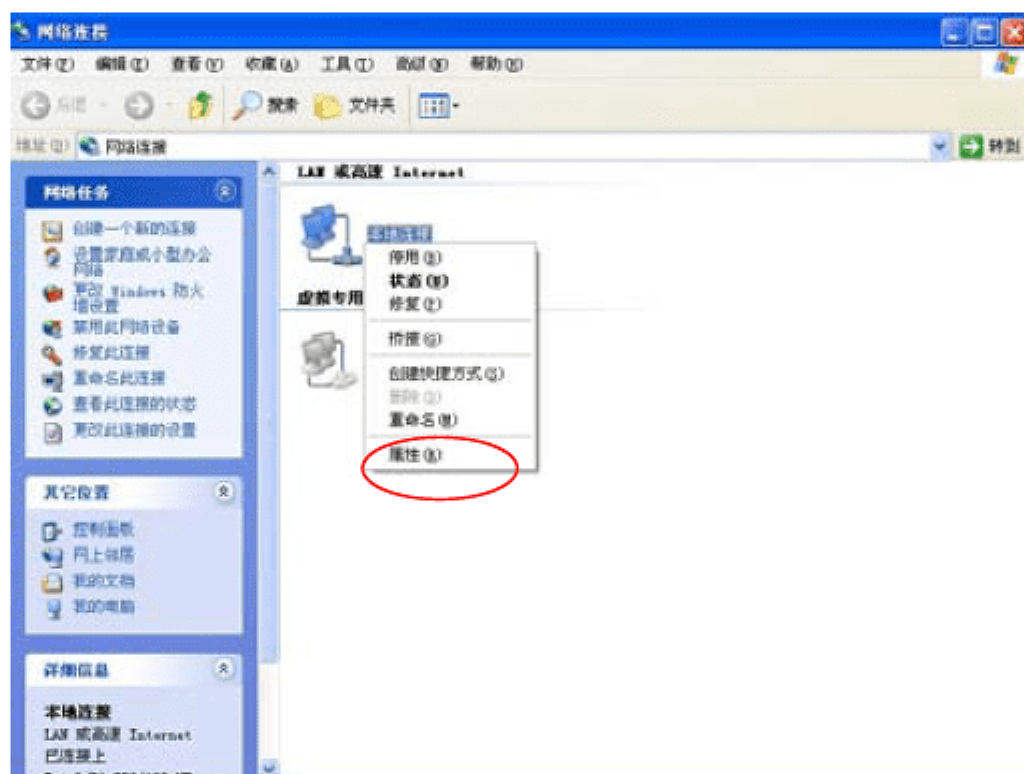


图 6-15　网络连接窗口

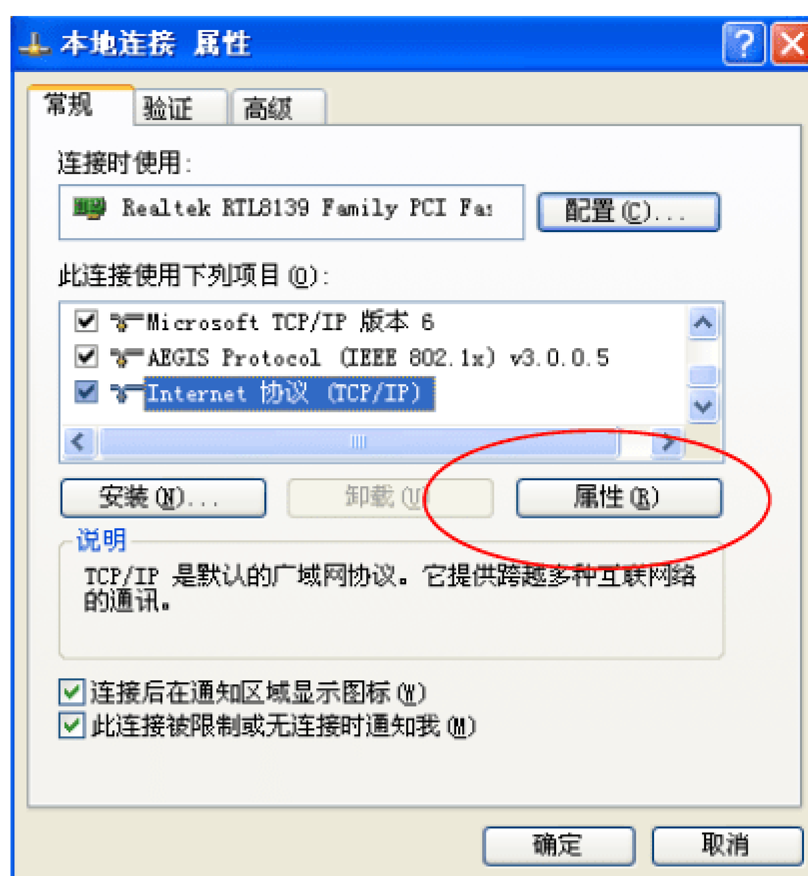


图 6-16　设置本地连接属性

❷ 在【Internet 协议(TCP/IP)属性】对话框中，选中【自动获得 IP 地址】和【自动获得 DNS 服务器地址】单选按钮，如图 6-17 所示。这也正是 DHCP 提供给用户的主要功能。

❸ 如何知道该台主机联网成功，即 DHCP 服务器成功分配 IP 给该客户端呢？可以通过查看客户端的 IP 和 DNS 地址来确定。选择【开始】→【运行】命令，弹出如图 6-18 所示的【运行】对话框，在【打开】文本框中输入 cmd，单击【确定】按钮就进入 DOS 命令行。

❹ 输入 ipconfig 命令，如果客户端已经联网会显示出基本的连网信息，如图 6-19 所示。可以看到，DNS Suffix 为 seu.edu.cn，正是我们前面在 DHCP 服务器端的/etc/dhcpd.conf 中设置的 option domain-name 选项。该客户端得到的动态 IP 为 172.18.12.239，确实在前面所设定的 IP 分配范围之内。这些都表示，DHCP 服务器设定成功，局域网内的客户端能顺利从 DHCP 服务器获得动态 IP。

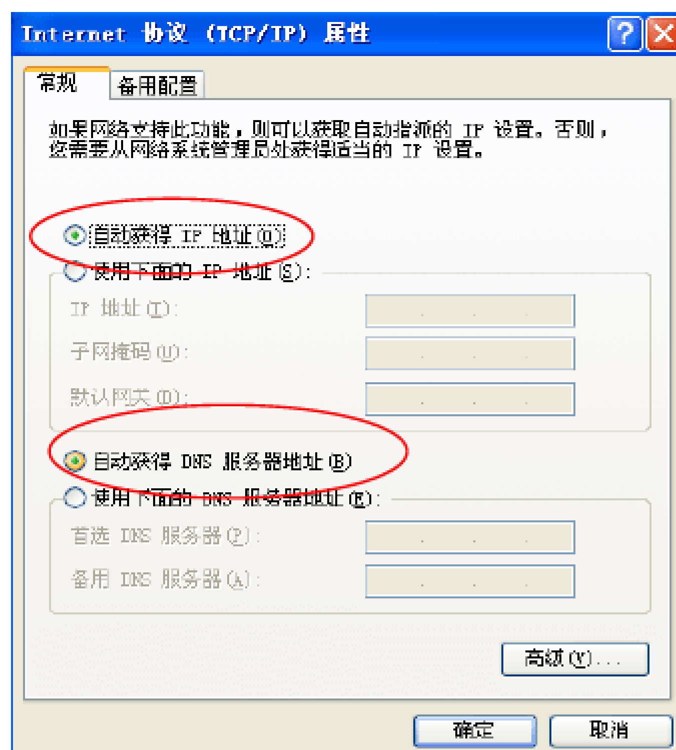


图 6-17　TCP/IP 设置

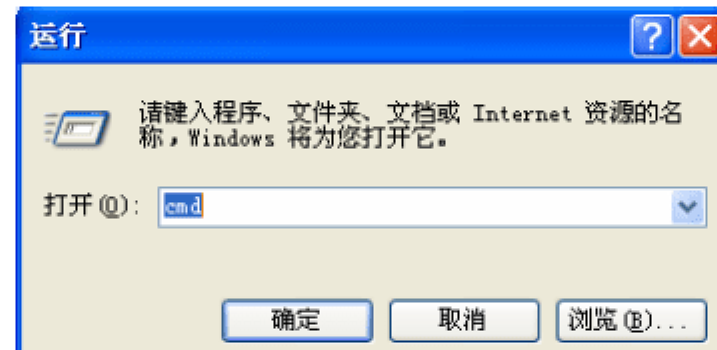


图 6-18　【运行】对话框

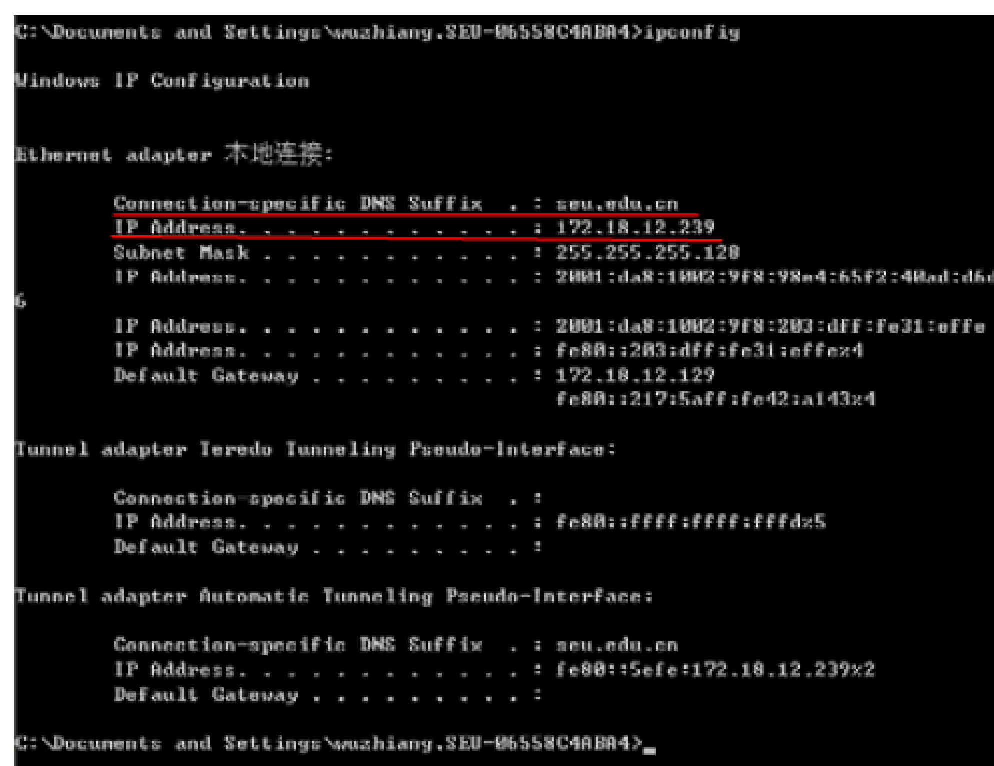


图 6-19　获得动态 IP 成功

❺ 如果要为客户端更新租约期限，可以使用下面命令，执行结果如图 6-20 所示。

```
ipconfig/renew
```

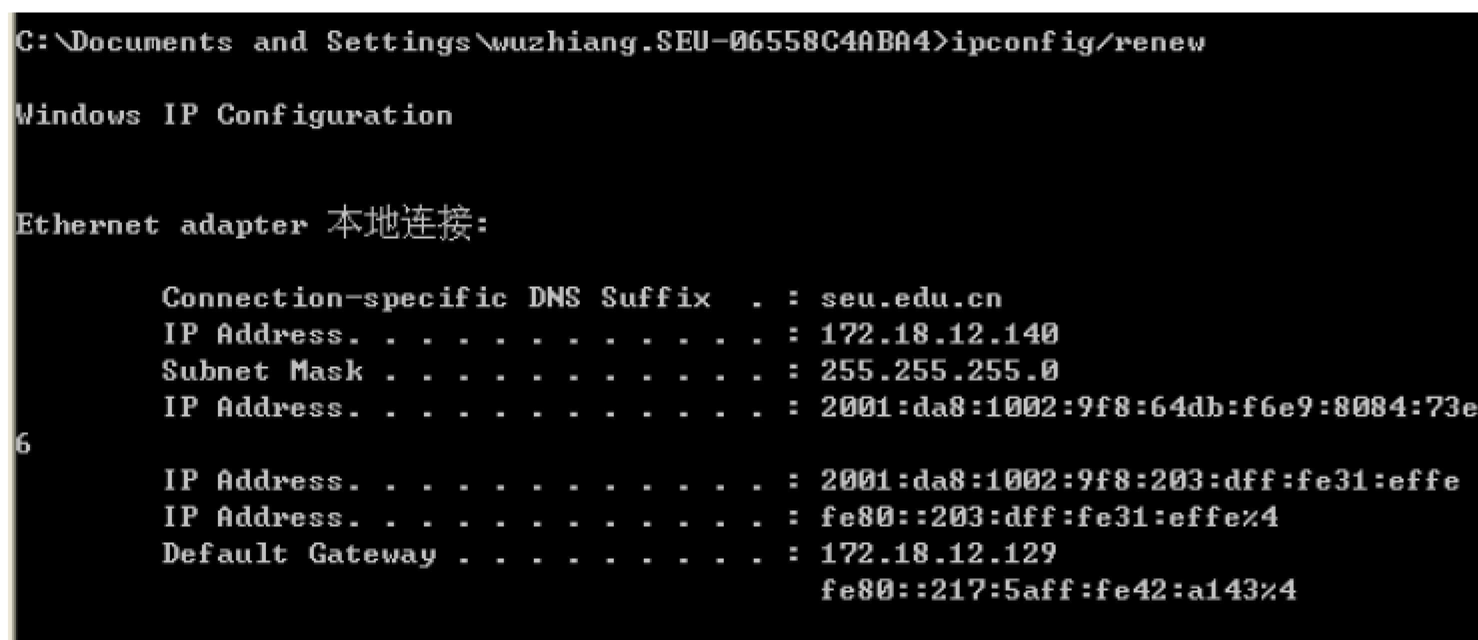


图 6-20　更新租约期限

点评与拓展：Windows 系统下查看网络配置的命令是 ipconfig，而 Linux 系统下实现同样功能的命令则为 ifconfig。

6.3.3　DHCP 静态 IP 的配置和测试

如果客户端因为断开连接或者租约期限到期，客户端都需要重新向 DHCP 服务器申请新的 IP 地址，那么客户端能否每次向 DHCP 服务器申请 IP 时都获得同样的 IP 地址呢？

当在局域网内架设 Web 服务器时这个问题就变得很有意义了。设想局域网内某台主机担任了 Web 服务器功能，它的 IP 地址就应该是固定的，IP 地址的随意变动将导致用户无法访问 Web 页面。这就要求 DHCP 服务器在这台客户端每次申请 IP 时，都分配给它固定的 IP。

本节在 Windows XP 客户端测试 DHCP 成功的基础上，深入讨论如何解决 DHCP 静态 IP 的问题。

❶ 在那台 Windows XP 的主机连接 DHCP 服务器，并获得 172.18.12.239 这个 IP 地址后，我们不妨打开/var/log/messages 日志文件看一看客户端的日志信息，如图 6-21 所示。

可以看到，这个文件记录了 DHCP 分配的所有 IP 地址的日志记录。图 6-21 中的标注部分记录了申请 172.18.12.239 这个 IP 地址的 MAC 地址和主机名，MAC 地址为 00:03:0d:31:ef:fe，主机名为 seu-06558c4aba4。我们知道 MAC 地址惟一确定了一块网卡，ARP(地址解析协议)建立了 MAC 地址和 IP 地址的映射关系，因此，基于这个思路，我们如果在 DHCP 服务器上将某个 MAC 地址与某个 IP 地址绑定在一起，就可以实现某客户端总能申请到固定的 IP 地址这个目的了。

```
[root@seugrid2 ~]# tail -n 15 /var/log/messages
Oct 14 19:39:53 seugrid2 dhcpd: dhcpd startup succeeded
Oct 14 19:47:28 seugrid2 dhcpd: DHCPDISCOVER from 00:03:0d:31:ef:fe via eth0
Oct 14 19:47:29 seugrid2 dhcpd: DHCPOFFER on 172.18.12.240 to 00:03:0d:31:ef:fe (seu-06558c4aba4) via eth0
Oct 14 19:47:33 seugrid2 dhcpd: DHCPDISCOVER from 00:03:0d:31:ef:fe (seu-06558c4aba4) via eth0
Oct 14 19:47:33 seugrid2 dhcpd: DHCPOFFER on 172.18.12.240 to 00:03:0d:31:ef:fe (seu-06558c4aba4) via eth0
Oct 14 19:47:40 seugrid2 dhcpd: DHCPDISCOVER from 00:03:0d:31:ef:fe (seu-06558c4aba4) via eth0
Oct 14 19:47:40 seugrid2 dhcpd: DHCPOFFER on 172.18.12.240 to 00:03:0d:31:ef:fe (seu-06558c4aba4) via eth0
Oct 14 19:47:40 seugrid2 dhcpd: DHCPREQUEST for 172.18.12.240 (172.18.12.178) from 00:03:0d:31:ef:fe (seu-06558c4aba4) vi
a eth0
Oct 14 19:47:40 seugrid2 dhcpd: DHCPACK on 172.18.12.240 to 00:03:0d:31:ef:fe (seu-06558c4aba4) via eth0
Oct 14 19:47:40 seugrid2 dhcpd: Abandoning IP address 172.18.12.240: declined.
Oct 14 19:47:40 seugrid2 dhcpd: DHCPDECLINE of 172.18.12.240 from 00:03:0d:31:ef:fe (seu-06558c4aba4) via eth0: not found
Oct 14 19:47:50 seugrid2 dhcpd: DHCPDISCOVER from 00:03:0d:31:ef:fe via eth0
Oct 14 19:47:51 seugrid2 dhcpd: DHCPOFFER on 172.18.12.239 to 00:03:0d:31:ef:fe (seu-06558c4aba4) via eth0
Oct 14 19:47:51 seugrid2 dhcpd: DHCPREQUEST for 172.18.12.239 (172.18.12.178) from 00:03:0d:31:ef:fe (seu-06558c4aba4) vi
a eth0
Oct 14 19:47:51 seugrid2 dhcpd: DHCPACK on 172.18.12.239 to 00:03:0d:31:ef:fe (seu-06558c4aba4) via eth0
```

图 6-21　记录客户端的日志信息

❷ 打开 DHCP 主配置文件/etc/dhcpd.conf，添加静态 IP 地址信息，如图 6-22 标注部分所示，解释如下：

```
host seu-06558c4aba4{
     hardware ethernet 00:03:0d:31:ef:fe;
     fixed-address 172.18.12.140;
  }
```

host 后面是主机名，我们将日志文件上记录的主机名写在此处。

harware ethernet 记录了 MAC 地址，fixed-address 记录了指定分配给该客户端的固定 IP

地址。这里我们将动态分配的 IP 范围规定在 172.18.12.100～172.18.12.240 内，并将 172.18.12.140 这个 IP 地址固定分配在 MAC 地址为 00:03:0d:31:ef:fe 的客户端。

```
#
# DHCP Server Configuration file.
#   see /usr/share/doc/dhcp*/dhcpd.conf.sample
#
ddns-update-style none;
default-lease-time 100000;
max-lease-time     200000;
option routers     172.18.12.129;
option broadcast-address 172.18.12.255;
option domain-name-servers 202.119.24.12;


subnet 172.18.12.0 netmask 255.255.255.0 {
   range 172.18.12.100 172.18.12.240;
   option subnet-mask 255.255.255.0;
   option nis-domain "seugrid2.seu.edu.cn";
   option domain-name "seu.edu.cn";

  host seu-06558c4aba4{

        hardware ethernet 00:03:0d:31:ef:fe;
        fixed-address 172.18.12.140;
    }

}
```

图 6-22　在 dhcpd.conf 中配置静态 IP

❸ 现在仍然在那台 Windows XP 的主机上测试静态 IP，将该主机的网卡停用或者使用下面命令释放 IP 地址，再重新连接，这样就能重新分配 IP。

Windows XP 释放 IP 地址命令:

```
ipconfig/release
```

成功获得 IP 后，我们在命令行输入 ifconfig 命令显示联网信息，如图 6-23 标注部分所示，这次获得的 IP 果然是 172.18.12.140，说明 DHCP 服务器静态 IP 设置成功。以后的每次连线，都将获得这个 IP 地址。

```
C:\Documents and Settings\wuzhiang.SEU-06558C4ABA4>ipconfig

Windows IP Configuration


Ethernet adapter 本地连接:

        Connection-specific DNS Suffix  . : seu.edu.cn
        IP Address. . . . . . . . . . . . : 172.18.12.140
        Subnet Mask . . . . . . . . . . . : 255.255.255.128
        IP Address. . . . . . . . . . . . : 2001:da8:1002:9f8:cc56:8330:2328:385
b
        IP Address. . . . . . . . . . . . : 2001:da8:1002:9f8:203:dff:fe31:effe
        IP Address. . . . . . . . . . . . : fe80::203:dff:fe31:effe%6
        Default Gateway . . . . . . . . . : 172.18.12.129
                                            fe80::217:5aff:fe42:a143%6
```

图 6-23　测试静态 IP

❹ 在 DHCP 静态 IP 设置中，经常需要知道其他主机的 MAC 地址，那么如何获得某个 IP 的 MAC 地址呢？这里介绍一个小技巧，即通过 ping 命令和 arp 命令的组合使用来获得，如图 6-24 标注部分所示，列出了上面 ping 命令所带参数的那个 IP 地址及其 MAC 地址。两条命令分列于下:

```
ping -c 1 172.18.12.179
arp -n
```

原理其实很简单，从 2.1 节“TCP/IP 协议族概述”中知道，TCP/IP 协议族中的 ARP 协议提供了从 IP 地址到 MAC 地址的映射转换功能。当收到由 IP 地址查询 MAC 地址的请求时，ARP 首先查找本地缓存中的 IP 地址和 MAC 地址的映射表，如果查不到对应项，则在所处网段内广播查询请求，与请求中 IP 地址一致的主机收到广播请求后将自己的 MAC 地址反向发出。ARP 主机收到所需要的 MAC 地址后，更新缓存中的映射表，这样下次查询该 IP 所对应的 MAC 地址时，就可以直接从缓存中获得，提高查询效率。因此，当输入 ping 命令时，主机获得该 IP 对应的 MAC 地址，并将其放置于缓存中，当我们接着输入 arp 命令查看缓存内容时，当然可以出现刚才 ping 那个 IP 的 MAC 地址了，MAC 地址是 48 位的二进制数，通常以十六进制的数显示出来，一共有 6 个分段，每个分段包括两个十六进制数字。

```
[root@seugrid2 ~]# ping -c 1 172.18.12.179
PING 172.18.12.179 (172.18.12.179) 56(84) bytes of data.
64 bytes from 172.18.12.179: icmp_seq=1 ttl=64 time=4.58 ms

--- 172.18.12.179 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 4.586/4.586/4.586/0.000 ms
[root@seugrid2 ~]# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
172.18.12.179            ether   00:0B:CD:CC:4B:8D   C                     eth0
172.18.12.225            ether   00:12:3F:C1:CF:2E   C                     eth0
[root@seugrid2 ~]#
```

图 6-24　获得某主机的 MAC 地址

6.4　本 章 小 结

DHCP 服务广泛地应用于局域网主机数量大或者移动设备接入频繁的环境中，它简化了网络管理员对 IP 地址的管理，有效避免了临时接入用户所产生的 IP 地址冲突问题，大大地方便了用户的接入。

本章主要介绍了 DHCP 的基本概念及工作原理、DHCP 服务器端及客户端的安装和配置等内容。分别介绍了 Linux 和 Windows 两种不同的客户端操作系统获得 DHCP 动态 IP 地址的配置，并且展开对 Windows 客户端的讨论，介绍并演示了配置静态 IP 地址的过程。读者需要着重理解 6.2 节、6.3.2 节和 6.3.3 节的应用实例，这样有助于理解 DHCP 动态 IP 分配、静态 IP 分配以及租约期等重要概念。

第 7 章　DNS 服务器的配置与架设

DNS(Domain Name System，域名系统)是 Internet 和 Intranet 中一种重要的网络服务，它可以将域名解析成 IP 地址，使用户只要记住简单的域名而不是由一连串数字所组成的 IP 地址即可访问网络。本章主要介绍了 DNS 的层次式域名结构、DNS 查询的基本原理，Cache-only 和 Forwarding 服务器，根 DNS 服务器及主从服务器的配置，以及 DNS 客户端的配置和测试等内容。

通过本章的学习，读者应掌握以下内容：

- ✧ DNS 的工作原理
- ✧ Cache-only 服务器的配置
- ✧ Cache-only 和 Forwarding 服务器的特点和意义
- ✧ 根 DNS 服务器的配置
- ✧ 主从 DNS 服务器的配置
- ✧ DNS 客户端的配置

7.1　DNS 服务概述

7.1.1　域名系统

DNS 是一种 TCP/IP 的标准服务，它是一种组织成域层次结构的计算机和网络服务命名系统，负责 IP 地址和域名之间的转换。第 2 章“Linux 服务器基本网络配置”中介绍过，TCP/IP 体系架构中网络层区分各个主机的方法是为每台主机分配一个惟一的 IP 地址来实现的。第 4 版本的 IP 地址是由 32 位二进制数组成的，将这 32 位二进制数分成 4 组，每组 8 个二进制数，然后将这 8 个二进制数转化成十进制数，就是普遍所看到的 IP 地址，其范围在 1～255 之间。但这些 IP 数字难以记忆，所以采用域名来取代数字，比如 Internet 用户能够很方便地记住www.google.com，但是很少的用户能记住 google 的 IP 地址。

在 Internet 上域名与 IP 地址之间是一一对应的，域名虽然便于 Internet 用户记忆，却不能作为网络层的惟一标识，本章所要讲述的 DNS 服务器就是为了实现域名到 IP 地址的映射转换的。DNS 服务器通过建立 DNS 数据库，记录主机名称与 IP 地址的对应关系，为客户端主机提供 IP 地址解析服务。当有主机要与网络上其他主机通信时，就可以通过域名向 DNS 服务器查询那台主机的 IP 地址。

域名系统需要维持全球所有域名与 IP 地址的对应关系，因此数据量是非常巨大的，为

了方便地管理这个海量数据库，域名系统采用了分布式分层管理的方式。DNS 服务器以两种标准将全世界的域名进行统一分类：第一种是以国家码，即这个域名属于哪个国家，由于互联网起源于美国的 Arpanet，因此不带国家码的域名默认是美国的域名，比如 www.google.cn 就说明这个域名是 google 在中国的镜像站，.cn 表示中国。第二种是以域名所属的机构类型来分类，主要有 6 大类，如表 7-1 所示。

表 7-1　DNS 域名系统分类

分类名称	描　述
com	公司、企业等
org	组织、机构
edu	大学等教育机构
gov	政府机关
net	网络、通信机构
mil	军事单位

DNS 的数据库结构是一个倒立的树状结构，最顶部是根域，根域下的是顶级域，顶级域一般有表 7-1 所示的 6 种类型以及所有国家码，每个国家码再划分出二级域，仍然是以此 6 种类型划分，因此一个国家的域名由以国家码所标识的顶级域进行统一分配和管理。二级域可以再划分出子域，子域下是主机或是再划分的子域，直至主机，如图 7-1 所示。整个域名地址从底部到根部逐层书写，中间以“.”隔开。

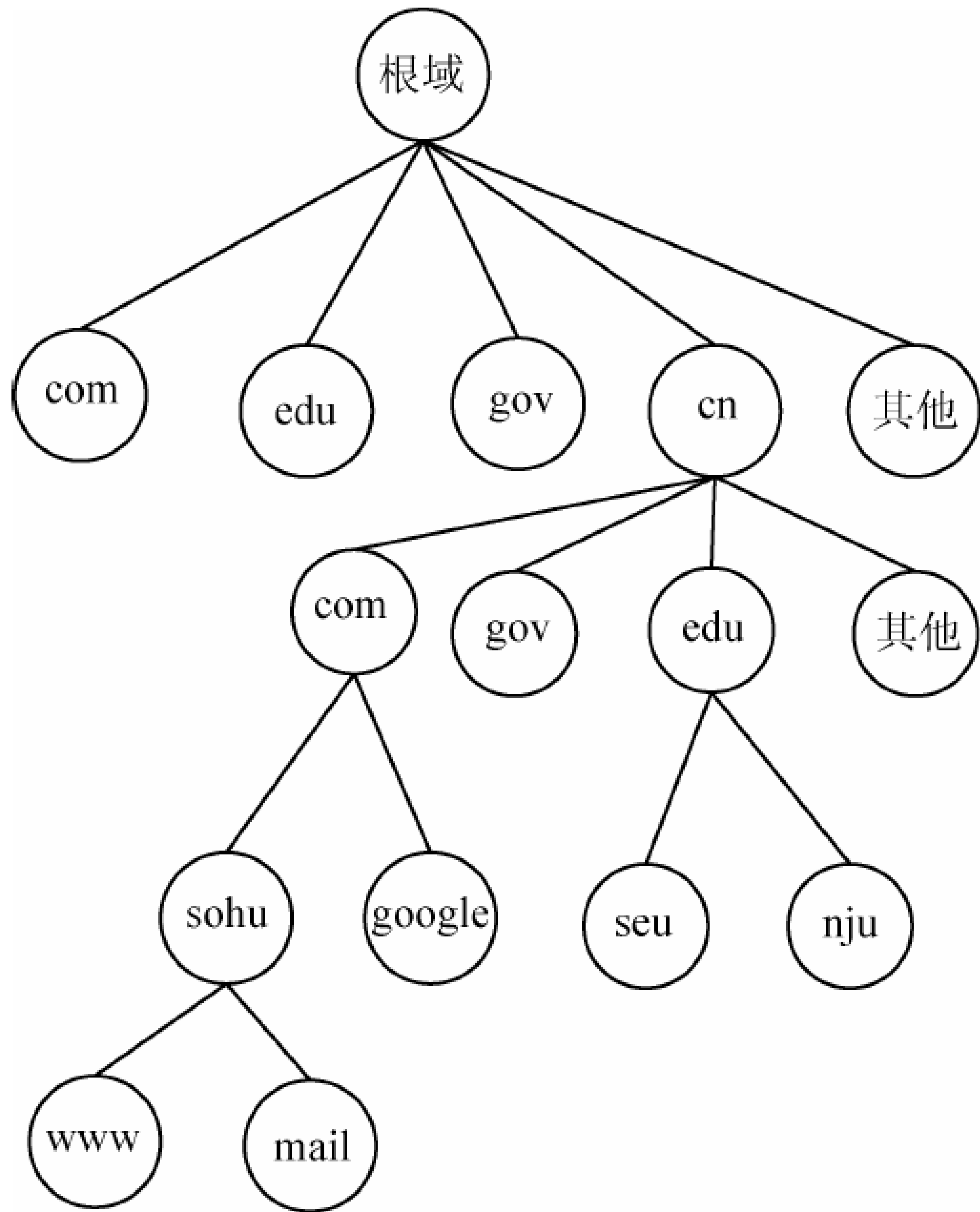


图 7-1　DNS 域名空间

7.1.2　DNS 的查询流程

客户端发送的查询消息包含以下 3 条：

✧ 指定的 DNS 域名，必须为完全符合标准的域名(FQDN)。

✧ 指定的查询类型，可根据类型指定资源记录，或者指定为查询操作的专门类型。

✧ DNS 域名的指定类型。

当要查询所使用的名称时，DNS 客户端会查询 DNS 服务器来解析。DNS 查询主要以下列 4 种方式进行解析：①客户端使用先前查询中获得的缓存信息就地应答查询；②DNS 服务器使用其自身的资源记录缓存信息来应答查询；③DNS 服务器代表客户端查询其他 DNS 服务器以完全解析该名称，然后将应答返回至客户端，即递归查询；④客户端本身查询其他 DNS 服务器来解析名称，并根据 DNS 服务器的应答使用其他的独立查询，即迭代查询。

DNS 查询过程一般分两步进行。

1．本地解析

名称查询首先从客户端计算机开始，先将解析请求传给本机的 DNS 客户服务程序。DNS 客户服务程序使用本地缓存信息进行解析，如果能够解析所查询的名称，则发送应答信息；如果与本机缓存中的信息不匹配，则解析过程继续，客户端将查询 DNS 服务器来进行解析。本地解析具体过程如图 7-2 所示。

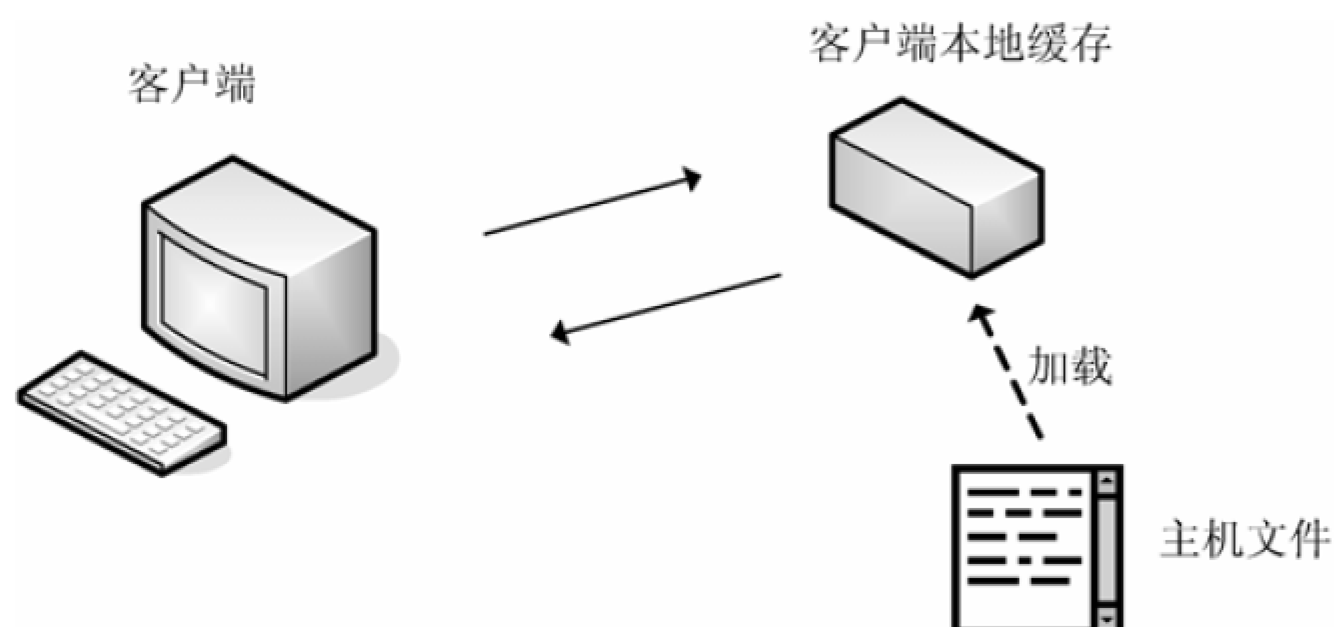


图 7-2　本地解析

本地缓存包括两种信息：①主机文件。该文件属本地配置，是主机名称到地址的映射信息，由 DNS 客户服务程序启动时预先加载至缓存。②资源记录。该信息是从先前的 DNS 查询的应答中取得的，将在缓存中保留一段时间。

2．查询 DNS 服务器

如果不能在本机解析查询，可将客户端请求发送至 DNS 服务器，通过查询 DNS 服务器来解析名称。DNS 服务器接到查询请求后，先查询能否在本地配置区域里获取匹配的资源记录信息。DNS 查询原理如图 7-3 所示。

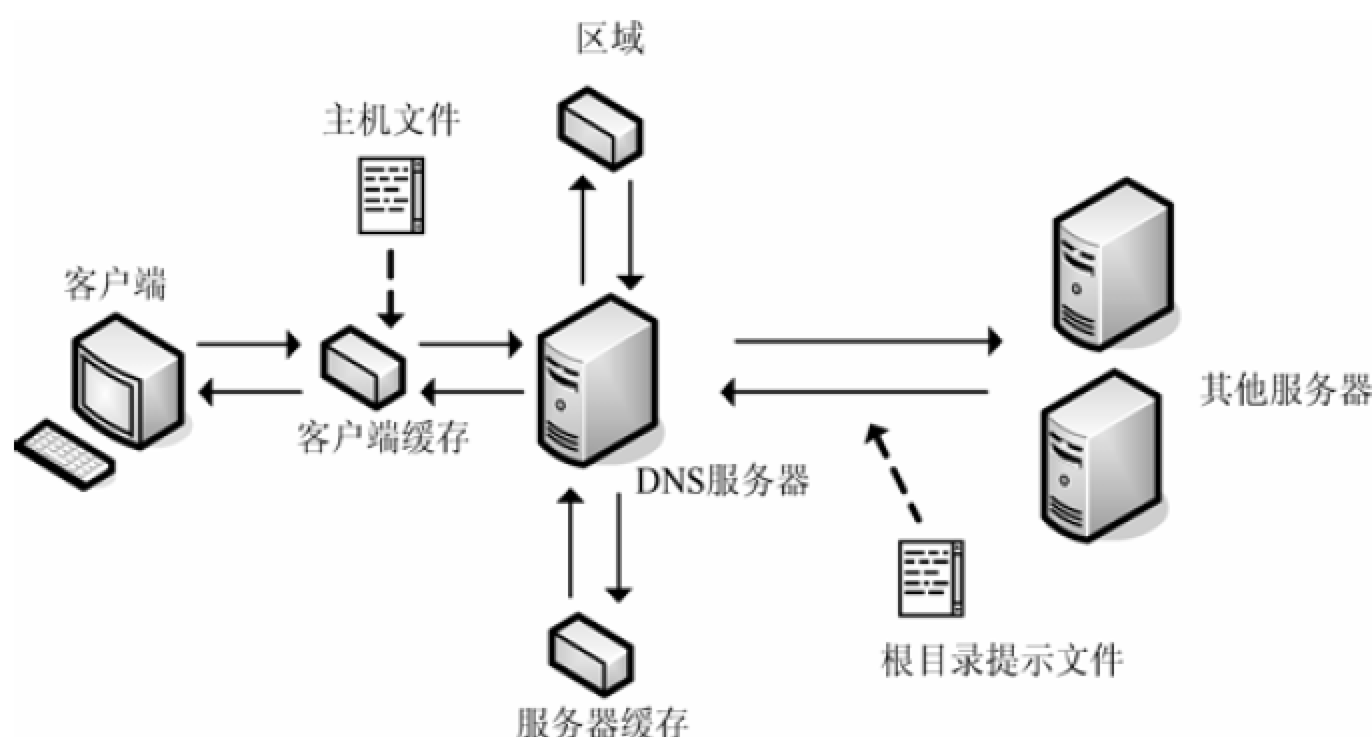


图 7-3　DNS 查询原理

如果所要查询的名称与本地区域中的相关资源记录信息完全匹配，则使用该信息来解析查询的名称，并发送应答信息。

如果本地区域中没有所要查询的名称的相关信息，则服务器尝试通过先前查询的本地缓存信息来解析名称。如果查询到匹配信息，则服务器使用该信息发出应答。

如果从缓存和本地区域中都未发现与查询的名称相匹配的信息，则查询过程继续，使用递归查询来完全解析名称。递归查询的工作流程如图 7-4 所示。

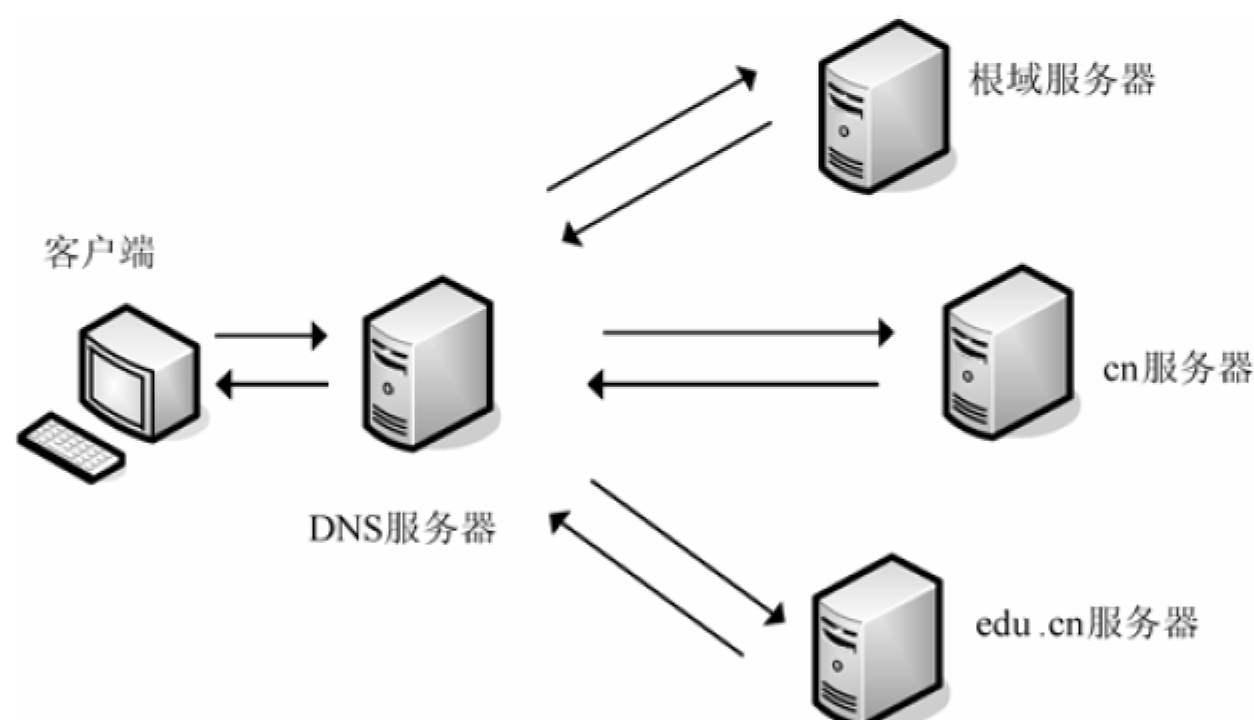


图 7-4　递归查询的工作流程

假设需要查询 seu.edu.cn 的地址，首选 DNS 服务器通过分析域名，向顶层域 cn 查询，而 cn 的 DNS 服务器与 edu.cn 的服务器联系以获取进一步的地址，这样循环查询直到获得需要的结果，然后一级一级向上返回查询结果，最终完成查询。此外，为了让 DNS 服务器可以正确进行递归查询，需要一些必要信息，该信息通常以根目录形式提供。借助根目录提示寻找根域服务器，DNS 服务器就可以完成递归查询。

如果 DNS 服务器上禁用递归查询或查询 DNS 服务器时客户端没有申请递归查询，则使用迭代查询。迭代查询的工作流程如图 7-5 所示。

假设现在要迭代查询 user.seu.edu.cn 的地址，而 DNS 服务器在本地查询不到客户端请求的名称信息时，就会代之以 DNS 客户端的身份向其他的 DNS 服务器继续进行查询以解

析该名称。一般情况下，查询搜索范围可能会一直扩展到根域服务器，但根域服务器并不进行完整应答，只返回 edu.cn 服务器的 IP 地址。DNS 服务器则根据该信息向 edu.cn 服务器进行查询，在 edu.cn 服务器完成对 user.seu.edu.cn 的域名解析后，再将结果返回 DNS 服务器。另外，对于大多数迭代查询，如果它的主 DNS 不能解析该名称，那么客户端就会使用本地配置的 DNS 服务器列表，在整个 DNS 域名空间中联系其他域名服务器。

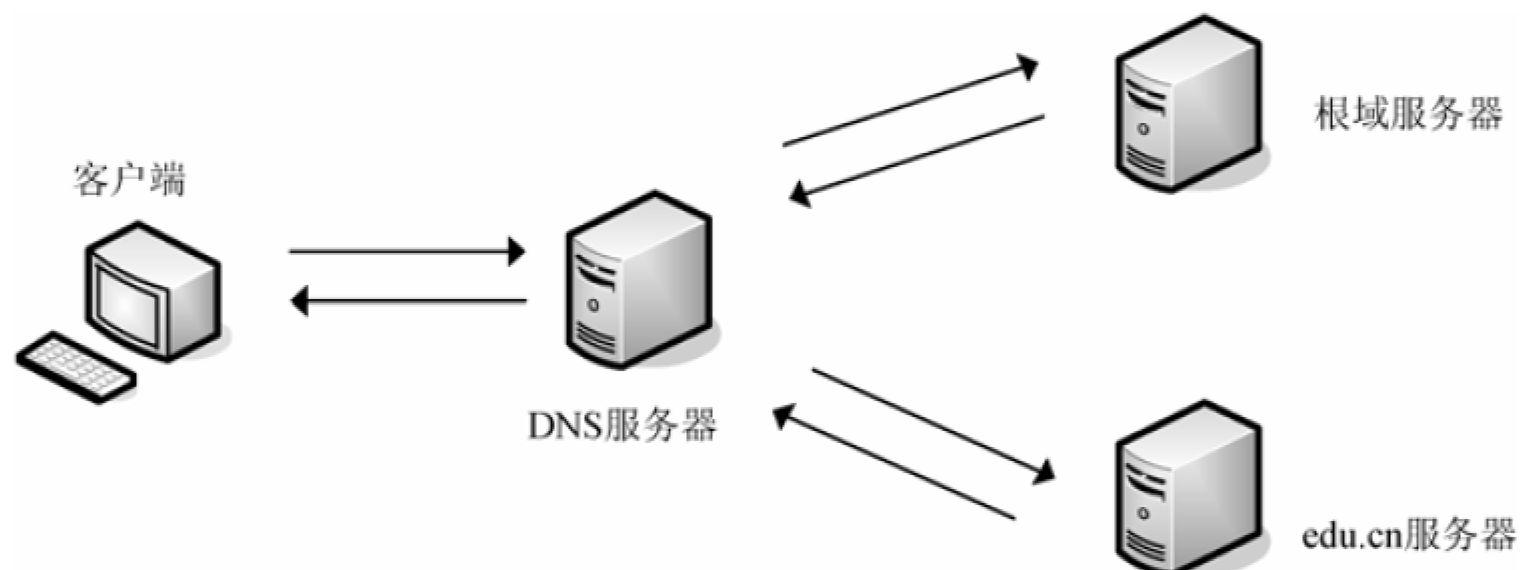


图 7-5　迭代查询的工作流程

7.1.3　正向解析与反向解析

正向解析是将域名映射为 IP 地址，例如，DNS 客户机可以查询主机名称为 www.seu.edu.cn 的 IP 地址。反向解析是将 IP 地址映射为域名。要实现反向解析，必须在 DNS 服务器中创建反向解析区域。反向域名的顶级域名是 in-addr.arpa。反向域名由两部分组成，域名前半段是其网络 ID 反向书写，而区域后半段必须是 in-addr.arpa。例如，要针对网络 ID 为 202.119.8.0 的 IP 地址来提供反向解析功能，则此反向区域的名称必须是 8.119.202.in-addr.arpa。

7.2　DNS 服务器端的配置

应用实例导航——Cache-only 服务器的架设

※场景呈现

A 公司准备架设一台 Cache-only 服务器，即在该服务器上不提供 DNS 数据库查询功能，只提供缓存和转发功能。

※技术要领

(1) Cache-only 和 Forwarding 服务器的概念和区别。
(2) Cache-only 服务器的配置。

DNS 服务器本身可以提供查询功能，也可能不提供查询功能，而仅提供转发功能。本节将分别介绍两类 DNS 服务器的服务器端配置方法。在此之前，首先介绍 Linux 系统下 DNS 软件所包括的一些组件，使读者对 DNS 有整体上的了解；然后介绍比较简单的 Cache-only DNS 服务器的配置，在此基础上，介绍 DNS 根服务器的配置方法。

7.2.1　DNS 软件结构简介

架设 DNS 服务器的常用软件是 BIND 程序，BIND 是加州大学伯克利分校开发出来的实现 DNS 服务器的开源软件，BIND 实际上是 Berkeley Internet Name Domain Service 的缩写。BIND 是由美国 DARPA 资助的研究项目，经过多年的不断发展，目前已经成为最广泛使用的 DNS 服务器软件。目前的最新版本为第 9 版，它修正了以前版本的很多错误，具有更高的稳定性和更高的执行效率，输入下面命令可以查看 BIND 的安装版本：

```
rpm -qa | grep bind
```

```
[root@sec ~]# rpm -qa | grep bind
bind-libs-9.3.2-4.1
bind-utils-9.3.2-4.1
bind-9.3.2-4.1
ypbind-1.19-0
kdebindings-3.5.1-1.2
bind-chroot-9.3.2-4.1
[root@sec ~]#
```

图 7-6　查看 BIND 版本

执行结果如图 7-6 所示，该服务器安装了 BIND 的 9.3.2 版本，bind-utils-9.3.2-4.1 是用于 DNS 客户端的软件，bind-9.3.2-4.1 是用于 DNS 服务器端的软件。

7.2.2　Cache-only 和 Forwarding DNS 服务器

1．Cache-only 和 Forwarding 的比较

Cache-only 和 Forwarding 服务器是最简单的两种 DNS 服务器，它们本身没有 DNS 域名数据库，也不能管理任何域，类似于代理服务器，它们都是将所收到的客户端的查询请求转发出去，然后等待其他 DNS 服务器的查询结果。

那么 Cache-only 和 Forwarding 两种服务器有何区别呢？首先，从名字上来看，Cache-only 服务器带了缓存机制，可以将最近用户的查询结果存放在缓存中，从而加快了客户端的响应速度；而 Forwarding 服务器未必要有缓存机制。其次，Forwarding 服务器可以将查询请求转发给任意的其他 DNS 服务器，也可能会转发到某台 Cache-only 服务器，但是 Cache-only 服务器不会将查询请求随便发送，而必定是转发到根 DNS 服务器查询，如图 7-7 所示，因此，Forwarding 服务器可以不知道根 DNS 服务器的地址，但是 Cache-only 却必须要知道。

两种服务器区别甚微，因此在配置方法上大同小异，或许命名为哪种服务器就取决于配置时是否设置缓存，转发的地址是否是根 DNS 服务器。

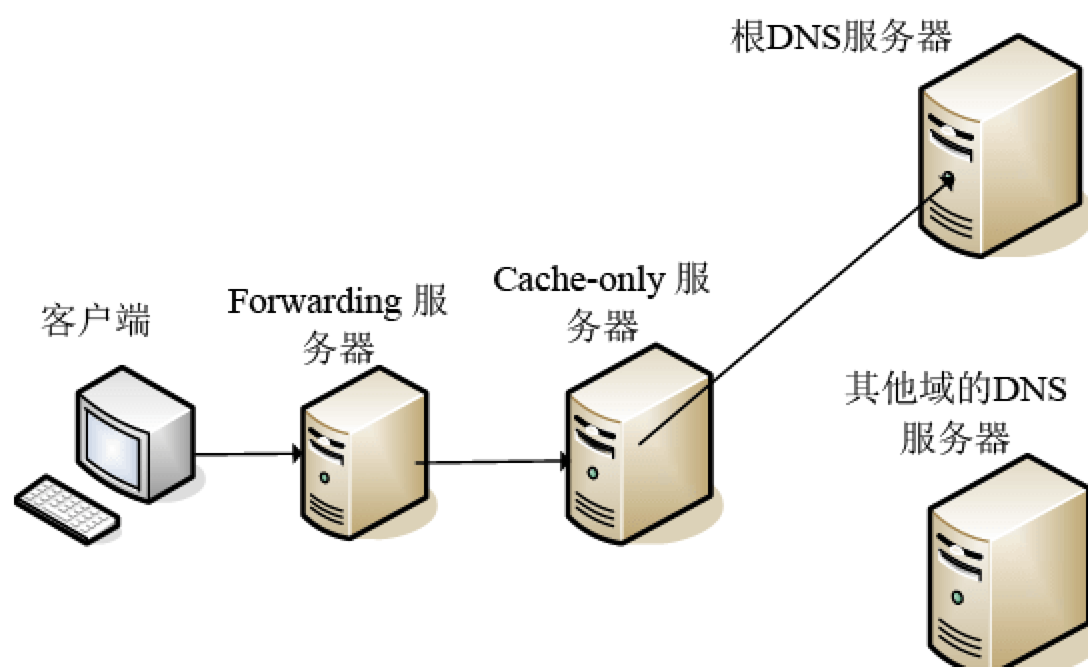


图 7-7　Cache-only 和 Forwarding 服务器示意图

2．Cache-only 服务器的配置

由于 Cache-only 服务器无需配置 DNS 数据库，也无需配置正反解域，只需配置/etc/named.conf 文件即可，因此相对简单些，本节就讲述如何配置 Cache-only 服务器。

从 DNS 软件结构中知道，DNS 的主配置文件是/etc/named.conf，所以配置 Cache-only 只需对该文件进行配置即可，使用 vi 命令打开文件，如图 7-8 所示。找到文件中的 options 一段，添加如图 7-9 标示部分的语句，解释如下：

```
[root@sec local]# vi /etc/named.conf
//
// named.conf for Red Hat caching-nameserver
//
```

图 7-8　打开/etc/named.conf

```
options {
        directory "/var/named";
        dump-file "/var/named/data/cache_dump.db";
        statistics-file "/var/named/data/named_stats.txt";

        pid-file "/var/run/named/named.pid";
        forward only;
        forwards {
              202.119.24.12;
              202.119.24.18;
        }
        /*
         * If there is a firewall between you and nameservers you want
         * to talk to, you might need to uncomment the query-source
         * directive below.  Previous versions of BIND always asked
         * questions using port 53, but BIND 8.1 uses an unprivileged
         * port by default.
         */
         // query-source address * port 53;
};
```

图 7-9　配置/etc/named.conf

```
options {
       directory "/var/named";
       dump-file "/var/named/data/cache_dump.db";
       statistics-file "/var/named/data/named_stats.txt";
      pid-file "/var/run/named/named.pid";
```

这三行采用默认值，pid-file 记录 named 服务的进程号。

```
       forward only;
       forwards {
             202.119.24.12;
             202.119.24.18;
        }
      };
```

forward only 说明 DNS 服务器只进行转发，不执行查询，这是 Cache-only 服务器最重要的设定。

forwards 里面记录了转发的地址，可以设为本机构的 DNS 服务器，或者设成 ISP 提供的 DNS 服务器。一般各个省市都有自己本地的服务器，比如江苏有两个：pub.jsinfo.net 221.228.255.2 和 ns.jsinfo.net 61.139.2.69。

我们可以总结出 named.conf 文件格式有如下规则。

每行语句必须以分号结尾。

如 options 这种的语句段，需要用花括号将设定的语句段包含起来。

注释符号与 C++语言类似，“/* */”符号注释一段语句，“//”符号注释一行语句。并且该文件也可以用 shell 脚本中的注释符号“#”，也是注释一行语句。

3．Cache-only 和 Forwarding 的深入讨论

从前面两种服务器的概念和配置中可以看出，Cache-only 和 Forwarding 服务器提供的功能很相似，本质上都是提供转发功能而并不提供查询功能。那么这种类型的服务器到底会给网络带来正面的还是负面的影响呢？也就是说，这种类型的服务器在网络中有没有必要存在呢？如果有必要存在，那么它们存在的意义在哪里呢？在本节中，我们就讨论一下这个有意思的话题。

对于 Cache-only 服务器的优劣，存在两种观点。第一种观点认为 Cache-only 起到了平衡负载的作用，如图 7-10 所示，当客户端很多，而根 DNS 只有一台时，Cache-only 通过缓存一些 DNS 数据库的记录可以满足很多客户端的查询要求，使得每台客户端不必都去访问根 DNS 服务器，从而减轻了根 DNS 服务器的负载，加快了对客户端的响应速度。第二种观点认为 Cache-only 和 Forwarding 服务器的这种转发功能增加了网络中 IP 包的数量，容易引起网络风暴。

实际上，两种观点都没有考虑到使用 Cache-only 服务器的网络环境，笔者认为，如果在用户兴趣相似、Web 服务器变动不频繁的环境中，Cache-only 服务器所缓存的 DNS 数据

库记录将很大概率地被再次使用，这时的 Cache-only 功能将大幅度加快对客户端的响应速度，平衡根 DNS 服务器的网络负载。例如，处于同一组织或公司的用户，由于工作需要很可能会访问相似的站点，这时一旦有一位用户访问过，记录就被存储在 Cache-only 服务器中，其他用户再次访问时就无需查询根 DNS 服务器，只需查询局域网内的 Cache-only 服务器即可获得该 DNS 记录。反之，如果用户查询的 DNS 记录大相径庭，Web 服务器又经常变更，那么 Cache-only 服务器的存在不仅不能带来实际的用处，还可能引起网络风暴。

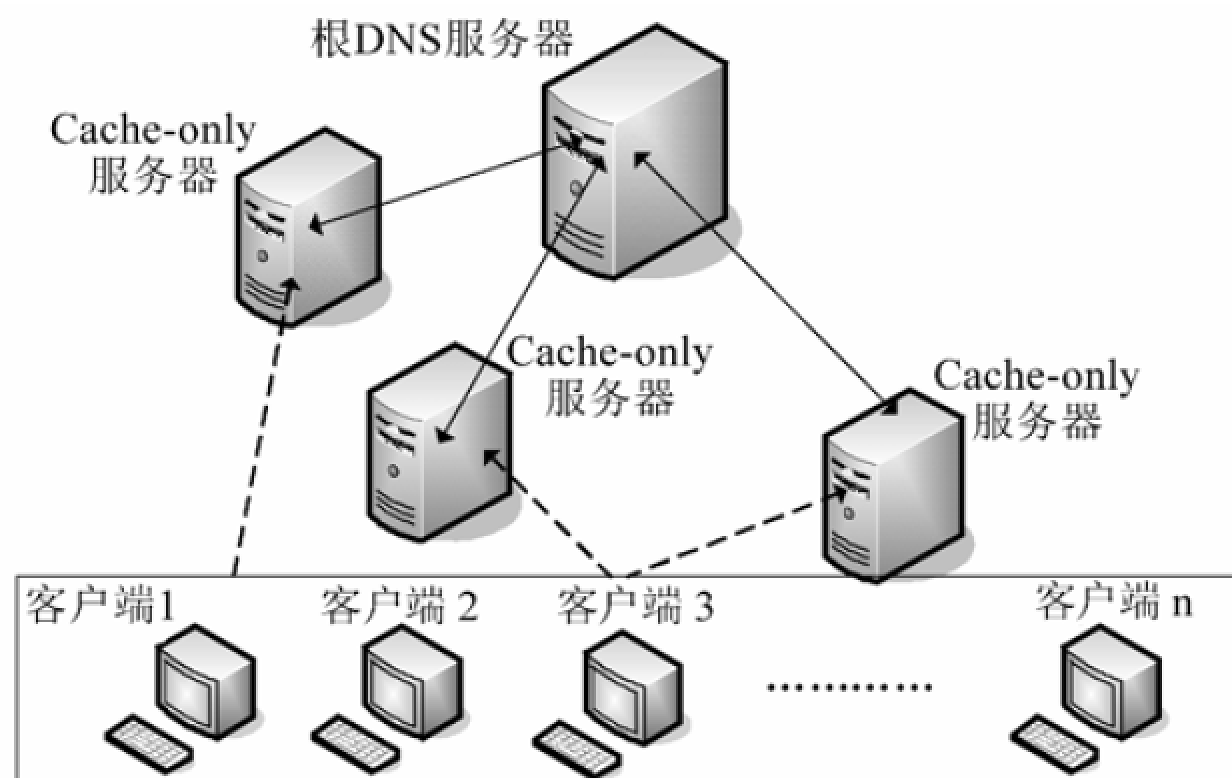


图 7-10　利用 Cache-only 服务器实现负载均衡

📖**点评与拓展**：简而言之，Cache-only 和 Forwarding 服务器与 DNS 根服务器的最大区别在于：这两种服务器不带 DNS 数据库文件，不提供查询功能，但是 Cache-only 有着类似于操作系统中缓存的作用，从而能够均衡网络负载，提高用户查询速度。

7.2.3　DNS 服务器端的配置

应用实例导航——DNS 主服务器的架设

※场景呈现

A 公司在成功架设一台 Cache-only 服务器后，进一步准备架设管理公司域的 DNS 服务器，所管理的域名为 seu.edu.cn。该 DNS 服务器需要提供该域内的所有主机的正向解析；该域的 IP 范围为 172.18.12.*，该 DNS 服务器也需要提供对这个范围内的 IP 地址的反向解析功能。

※技术要领

(1) DNS 服务器的 chroot 和管理域设置。
(2) DNS 服务器正向解析数据库文件的创建。

(3) DNS 服务器反向解析数据库文件的创建。

在对 Cache-only 这种简单的 DNS 服务器有所了解后，本节介绍 DNS 根服务器的配置，这类服务器提供查询库，维持了 DNS 数据库，并提供对管理域内主机的正向和反向解析功能。根 DNS 服务器的配置文件主要有三个：根 DNS 服务器的主配置文件、正向解析配置文件以及反向解析配置文件。

1．根 DNS 服务器的设置

根 DNS 服务器的主配置文件依然是/etc/named.conf 文件，需要在此文件中设置是否提供查询功能、是否提供转发功能、正向解析配置数据库文件的路径和域名、反向解析配置数据库文件的路径和 IP 段。

❶ 首先设定/etc/named.conf 的 options 部分，这里规定了 DNS 服务器的权限，是 DNS 服务器的总设定。添加如图 7-11 标注部分所示的语句。前面四行与 Cache-only 服务器的设定相同，在此不作详细解释，我们仅解释最后三行:

```
forwarders {202.119.24.12; 202.119.24.18;};
```

指定了转发地址，即如果在本地 DNS 数据库查询不到，就将用户查询请求转发到此处设定的地址。

```
allow-query {any;};
```

设定是否允许查询，此项是与 Cache-only 服务器本质不同的地方，根服务器当然是允许的了。

```
allow-transfer {none;};
```

是否允许传送域，默认不可以，通常不作修改。

```
[root@sec local]# vi /etc/named.conf
//
// named.conf for Red Hat caching-nameserver
//

options {
        directory "/var/named";
        dump-file "/var/named/data/cache_dump.db";
        statistics-file "/var/named/data/named_stats.txt";
        pid-file "/var/run/named/named.pid";

        forwarders {202.119.24.12; 202.119.24.18;};
        allow-query {any;};
        allow-transfer {none;};
        /*
         * If there is a firewall between you and nameservers you want
         * to talk to, you might need to uncomment the query-source
         * directive below.  Previous versions of BIND always asked
         * questions using port 53, but BIND 8.1 uses an unprivileged
         * port by default.
         */
         // query-source address * port 53;
};
```

图 7-11　DNS 服务器权限设定

❷ 然后设置在管理域内如何作正向解析和反向解析，添加如图 7-12 所示的语句。可以看到实际上是添加了两个 zone 结构，一个用于正解，一个用于反解。下面分别解释这两个 zone 的意义。

```
zone "seu.edu.cn" {
        type master;
        file "named.seu.edu.cn";
};
```

这个 zone 用于正向解析，我们知道正向解析是从域名查找出它所对应的 IP，因此正向解析的 zone 必须设置域名，本例 DNS 管理的域的域名是 seu.edu.cn。

type 是该 zone 的类型，DNS 服务器一共规定了三种类型的 zone：master、slave 和 hint。其中最上层的 DNS 服务器用 hint 类型，DNS 主服务器使用 master 类型，从服务器使用 slave 类型。这个服务器是 seu.edu.cn 域内的主服务器，因此使用 master 类型。

file 设置了正向解析数据库文件的相对路径，那么 DNS 服务器如何获得该文件的绝对文件路径呢？前面的 chroot 和 options 里的 directory 选项，正是由这两个参数拼装出了文件的绝对路径，格式如下：[chroot 目录]/[options 中的 directory 目录]/[file 设定值]。本例中的绝对路径为/var/named/chroot/var/named，至于如何建立 named.seu.edu.cn 这个文件将在 7.3 节介绍，但是启动 named 服务之前，一定要保证/var/named/chroot/var/named 这个目录中有 named.seu.edu.cn 这个文件。

```
zone "12.18.172.in-addr.arpa" IN {
        type master;
        file "named.172.18.12";
};
```

这个 zone 用于反向解析，反向解析是从 IP 查找出它所对应的域名，因此反向解析的 zone 必须设置 IP 段，本例将 IP 段设为 172.18.12 的意思是反向适用于 172.18.12.*这个范围内的 IP 地址。

type 同样为 master。

file 定义了反向解析数据库文件名为 named.172.18.12，同样启动 named 服务之前，一定要保证/var/named/chroot/var/named 这个目录中有 named.172.18.12 这个文件。

```
zone "seu.edu.cn" {
        type master;
        file "named.seu.edu.cn";
};

zone "12.18.172.in-addr.arpa" IN {
        type master;
        file "named.172.18.12";
};
```

图 7-12　/etc/named.conf 中的正反向解析设置

2．DNS 服务器正向解析数据库文件的建立

DNS 的查询依赖于正向解析数据库文件，当用户 DNS 查询请求到达后，首先查询这些正向解析数据库文件，如果是最上层的 DNS 服务器，它保存的数据库文件记录了次层所有

主机和 IP 的对应关系，官方 ISP 提供了这种数据库文件。本节所讲述的数据库文件仅记录了本管理域内的域名与 IP 地址的对应关系，但与官方提供的最高层的 root 服务器的数据库文件格式一致，设定方法也一致。下面讲述如何建立正向解析数据库文件，并解释其含义。

❶ 使用 vi 打开/var/named/chroot/var/named/named.seu.edu.cn 文件，添加如图 7-13 所示的全部语句。这两段包含了 TTL 和 SOA 的设置。

```
$TTL    600
```

该项设定了 TTL 时间，何谓 TTL 呢？前面概述部分讲过当 DNS 成功查询到某个域名所对应的 IP 后，通常会将此结果放在本地一段时间，TTL 就是规定将结果放置多长时间的选项。它设定了当外部 DNS 服务器查询本域的主机时，结果放置在那台外部 DNS 服务器中的时间，单位为秒。

TTL 如何设定才合适呢？一般来说，如果本域内的主机的对应关系变动频繁，TTL 宜设置得小些；如果本域内的主机的对应关系长期固定不变，TTL 可以设得大些。本例设为 600 秒。

```
@ IN SOA seugrid3.seu.edu.cn. zawu.seu.edu.cn. (
                        2007101801
                        28 800
                        14 400
                        720 000
                        86 400
                        )
```

这段语句是关于 SOA 的设定，何谓 SOA 呢？SOA 是 Start of Authority 的缩写，它是主 DNS 服务器必须要设定的选项，记录了这个 DNS 服务器是主 DNS 服务器，并定义了域名数据库文件的各种属性。

@ IN SOA 是固定不变的，说明开始了 SOA 设定；接着就是设定 DNS 服务器的主机名，即本机的域名，注意名字后要加句号“.”，这个句号经常容易被忽略，一旦忽略 named 服务将无法启动，本例中设为 seugrid3.seu.edu.cn.，如图 7-13 标注部分所示。然后是设定管理员 E-mail，由于@符号已经另作他用，因此这里用“.”代替@符号，读者也可以根据实际情况设置，无特殊要求。括号中设置了五组数字，第 1 组数字 2007101801 是该数据库文件的序列号，读者可以任意设置；第 2 组数字用于开放从服务器的情况，表示隔多久时间对从服务器更新一次；第 3 组数字表示，当到了更新时间，如果连接不上从服务器，隔多久再次尝试与从服务器的连接；第 4 组数字表示，当一直连接不上从服务器，隔多久时间之后，不再连接从服务器，而判定无法连接从服务器；第 5 组数字与 TTL 功能一样，当没有设定 TTL 时，这个数值就作为 TTL。

❷ SOA 设置完毕后，就可以建立域名与 IP 的映射关系了。首先是设定 DNS 服务器的名称，格式如下：[zone] IN NS [主机名称]，如果有多个 DNS 服务器主机，这里就分多行设定，如图 7-14 的第一行所示。

接着按照格式：[主机名] IN A [IP 地址]，逐行写出本域内所有主机和 IP 的对应关系。由于从前面的设定已经能够判定域名是 seu.edu.cn 了，因此此处只要填入主机名即可，如图 7-14

后面三行所示。

```
[root@sec local]# vi /var/named/chroot/var/named/named.seu.edu.cn
$TTL    600
@ IN SOA seugrid3.seu.edu.cn. zawu.seu.edu.cn. (
                            2007101801
                            28800
                            14400
                            720000
                            86400
                            )
```

图 7-13　正向解析数据库文件的 SOA 设定

```
@           IN    NS      seugrid3.seu.edu.cn.
seugrid3    IN    A       172.18.12.179
seugrid2    IN    A       172.18.12.178
seugrid1    IN    A       172.18.12.181
```

图 7-14　设定域名与 IP 的映射关系

点评与拓展：再次提醒读者，在正解数据库文件中所有出现主机名和 E-mail 的地方，最后都要加上句号 "."，这与我们一般的习惯不一样，容易被忽略。如果没有句号，DNS 就是用其相对名称，在后面自动加上域名 seu.edu.cn，比如本例中 DNS 将把 seugrid3.seu.edu.cn(不带句号)理解为 seugrid3.seu.edu.cn.seu.edu.cn。

3．DNS 服务器的反向解析数据库文件

正向解析数据库文件中记录了域名和 IP 的对应关系，反向解析数据库文件则记录了 IP 和域名的对应关系。下面讲述本例中如何创建 named.172.18.12 文件，以实现在 172.18.12.* 网段内的反解。

❶ 反向解析数据库文件 TTL 和 SOA 的设定方法与正向解析数据库文件一样，只是序列号不能设为相同，至于语句含义在此不再详述，请参考上一节步骤，如图 7-15 所示。

```
[root@sec local]# vi /var/named/chroot/var/named/named.172.18.12
$TTL    600
@ IN SOA seugrid3.seu.edu.cn. root.seu.edu.cn. (
                            2007101813
                            28800
                            14400
                            720000
                            86400
                            )
```

图 7-15　反向解析数据库文件的 SOA 设定

❷ 建立 IP 与域名的反向映射关系时，同样以如下格式：[zone] IN NS [主机名称]，先设定 DNS 服务器，如图 7-16 第一行所示。然后按照格式：[主机名] IN PTR [IP 地址]，逐行写出本 IP 段内所有 IP 和主机的对应关系，如图 7-16 所示，可以看到这里仅写了 IP 的最后一个分位，因为我们在/etc/named.conf 这个配置文件中已经设定好反向解析的网段了，即 172.18.12。如果/etc/named.conf 的设定为 172.18，那么此处就需要写入 IP 地址的最后两个分位。

```
@            IN    NS    seugrid3.seu.edu.cn.
178          IN    PTR   seugrid2.seu.edu.cn.
181          IN    PTR   seugrid1.seu.edu.cn.
~
~
~
```

图 7-16　设定 IP 与域名的映射关系

4．DNS 服务器的启动与观察

在对 DNS 服务器端设置完毕后，本节讲述如何启动 named 服务，及如何确定 named 服务成功启动。

❶ 输入下面命令启动 named 服务，如图 7-17 所示。关闭、重启 named 服务，查看 named 服务状态等命令格式亦列于下方。

```
/etc/init.d/named start
/etc/init.d/named  {stop|restart|reload|condrestart|status}
```

```
[root@sec local]# /etc/init.d/named start
启动 named:                                              [确定]
```

图 7-17　启动 named 服务

❷ DNS 服务的默认端口是 53，输入下面命令查看是否有启动在 53 号端口的监听进程。图 7-18 中的标注部分显示了两个进程，这两个进程就是 named 服务。

```
netstat -utln | grep 53
```

```
[root@sec local]# netstat -utln | grep 53
tcp        0      0 172.18.12.179:53            0.0.0.0:*                   LISTEN
tcp        0      0 127.0.0.1:53                0.0.0.0:*                   LISTEN
tcp        0      0 127.0.0.1:953               0.0.0.0:*                   LISTEN
tcp        0      0 ::1:953                     :::*                        LISTEN
udp        0      0 172.18.12.179:53            0.0.0.0:*
udp        0      0 127.0.0.1:53                0.0.0.0:*
udp        0      0 0.0.0.0:5353                0.0.0.0:*
```

图 7-18　named 服务监听进程

❸ 此时再看看日志信息，输入下面命令查看/var/log/messages 日志文件的最后 15 行，如图 7-19 所示。可以看到第一个标注部分显示了 named 服务在哪几块网卡上监听，第二个标注部分记录了 named 服务所载入的 zone 数据库文件，再看看序列号，正是刚才所创建的 named.seu.edu.cn 和 named.172.18.12 文件。

```
tail -n 15 /var/log/messages
```

📖**点评与拓展：**named 服务的启动类似于 dhcpd 服务，并不是启动时显示 OK 就表示服务启动成功，需要从端口监听进程和日志文件来观察。另外，日志文件还会提供 named 服务启动不成功时的错误信息。

```
[root@sec local]# tail -n 15 /var/log/messages
Oct 18 13:15:33 sec named: zone seu.edu.cn/IN: loading master file named.seu.edu.cn: file not found
Oct 18 13:15:33 sec named: _default/seu.edu.cn/IN: file not found
Oct 18 13:15:33 sec named: zone 12.18.172.in-addr.arpa/IN: loading master file named.172.18.12: file not found
Oct 18 13:15:33 sec named: _default/12.18.172.in-addr.arpa/IN: file not found
Oct 18 13:19:08 sec named[17555]: starting BIND 9.3.2 -u named -t /var/named/chroot
Oct 18 13:19:08 sec named[17555]: found 2 CPUs, using 2 worker threads
Oct 18 13:19:08 sec named[17555]: loading configuration from '/etc/named.conf'
Oct 18 13:19:08 sec named[17555]: listening on IPv4 interface lo, 127.0.0.1#53
Oct 18 13:19:08 sec named[17555]: listening on IPv4 interface eth0, 172.18.12.179#53
Oct 18 13:19:08 sec named[17555]: command channel listening on 127.0.0.1#953
Oct 18 13:19:08 sec named[17555]: command channel listening on ::1#953
Oct 18 13:19:08 sec named[17555]: named.172.18.12:6: file does not end with newline
Oct 18 13:19:08 sec named[17555]: zone 12.18.172.in-addr.arpa/IN: loaded serial 2007101813
Oct 18 13:19:08 sec named[17555]: zone seu.edu.cn/IN: loaded serial 2007101801
Oct 18 13:19:08 sec named[17555]: running
[root@sec local]#
```

图 7-19　named 服务日志信息

7.3　DNS 主从服务器的配置

应用实例导航——A 公司 DNS 主从服务器的架设

※场景呈现

A 公司架设好根 DNS 服务器后，发现一台 DNS 服务器负载太重，不能快速响应用户的 DNS 查询请求，需要配置一台从服务器以减轻主 DNS 服务器的负担。

从服务器的 IP 地址为 172.18.12.146，要求同样能够实现网段 172.18.12.*中的正向解析和反向解析功能。

※技术要领

(1) 主服务器开放从服务器权限。
(2) 从服务器的配置和启动。

7.3.1　主 DNS 服务器权限的开放

首先需要在主 DNS 服务器上开放从 DNS 服务器权限，通过设置/etc/named.conf 允许 zone 资料的传送，还要在正反解数据库文件中添加从 DNS 服务器的相关信息。具体主服务器端的配置步骤如下。

❶ 打开/etc/named.conf 文件，找到正反解 zone 的设置部分，添加 allow-transfer 语句，如图 7-20 所示。allow-transfer 可以带多个从 DNS 服务器地址参数，中间用封号隔开。本例只设置一个地址，因此语句为：

```
allow-transfer{172.18.12.146;};
```

从服务器如果要提供查询功能，必须要有 DNS 服务器的正向解析数据库文件和反向解析数据库文件，而这两个文件又必须与主服务器一样，因此，通常直接从主服务器将这两个文

件传送到从服务器的 chroot 目录下。上面这条语句就是允许主服务器将正反解的 zone 文件传送到地址为 172.18.12.146 的从服务器。

```
zone "seu.edu.cn" {
        type master;
        file "named.seu.edu.cn";
        allow-transfer {172.18.12.146;};
};

zone "12.18.172.in-addr.arpa" IN {
        type master;
        file "named.172.18.12";
        allow-transfer {172.18.12.146;};
};
```

图 7-20　allow-transfer 设定

❷ 在正向解析数据库文件中添加从服务器地址和主机名相关内容，如图 7-21 标注部分所示。格式与设定主服务器一样，以 NS 为标记。

```
[root@sec ~]# vi /var/named/chroot/var/named/named.seu.edu.cn
$TTL    600
@ IN SOA seugrid3.seu.edu.cn. zawu.seu.edu.cn. ( 2007101801 2880

@           IN   NS     seugrid3.seu.edu.cn.
@           IN   NS     cgsp.seu.edu.cn.
seugrid3    IN   A      172.18.12.179
cgsp        IN   A      172.18.12.146

seugrid2    IN   A      172.18.12.178
seugrid1    IN   A      172.18.12.181
```

图 7-21　正解数据库文件设定

❸ 在反向解析数据库文件中作类似的设置，如图 7-22 标注部分所示。

```
$TTL    600
@ IN SOA seugrid3.seu.edu.cn. root.seu.edu.cn. ( 2007101813 28

@           IN   NS     seugrid3.seu.edu.cn.
@           IN   NS     cgsp.seu.edu.cn.
179         IN   PTR    seugrid3.seu.edu.cn.
146         IN   PTR    cgsp.seu.edu.cn.
178         IN   PTR    seugrid2.seu.edu.cn.
181         IN   PTR    seugrid1.seu.edu.cn.
```

图 7-22　反解数据库文件设定

7.3.2　从服务器的配置

从服务器端的设置很简单，因为不涉及正反向解析数据库文件的设定，只需在/etc/named.conf 中作简单设定即可完成，具体步骤如下。

❶ 登录 172.18.12.146 这台主机，打开/etc/named.conf 文件，DNS 服务器的总设定部分与根服务器类似，所不同的是 zone 部分的设定。在 7.2.3 节中讲到 zone 的类型时，其中包含一种

类型叫 slave 类型，这就是指定了服务器为从 DNS 服务器，因此从服务器的 type 就设定为 slave，并且相应地设定主服务器的地址，如图 7-23 标注部分所示。

本例的设置表示该台服务器为从服务器，主服务器的地址是 172.18.12.179，正向解析数据库的文件为 named.seu.edu.cn，反向解析数据库文件为 named.172.18.12。

```
zone "seu.edu.cn" {
        type slave;
        file "named.seu.edu.cn";
        masters {172.18.12.179;};
};

zone "12.18.172.in-addr.arpa" IN {
        type slave;
        file "named.172.18.12";
        masters {172.18.12.179;};
};
```

图 7-23　反解数据库文件设定

❷ 对/etc/named.conf 设置完毕后，就可以启动从服务器的 named 服务，如图 7-24 所示。

```
[root@cgsp ~]# /etc/init.d/named start
Starting named:                                            [  OK  ]
[root@cgsp ~]#
```

图 7-24　启动从服务器

❸ 从服务器端 named 服务启动之后，会自动向主 DNS 服务器端发请求复制正反解数据库文件，如果过一段时间能在/var/named/chroot/var/named 目录中看到 named.seu.edu.cn 和 named.172.18.12 两个文件，则表示从服务器配置和启动成功，如图 7-25 标注部分所示。

```
[root@cgsp ~]# ls /var/named/chroot/var/named/
data             localhost.zone  named.broadcast  named.ip6.local  named.seu.edu.cn  slaves
localdomain.zone named.172.18.12 named.ca         named.local      named.zero
[root@cgsp ~]#
```

图 7-25　传送来的正反解文件

❹ 如果在/var/named/chroot/var/named 目录长时间看不到正反解数据库文件，则查看/var/log/messages 日志文件，可能会出现如图 7-26 标注部分所示的错误。那么再次查看主从 DNS 服务器端的配置，看是否有配置文件的设定错误；确定配置无误后，还需要查看防火墙设定，看 53 号端口是否被关闭。

点评与拓展：主从 DNS 服务器设定的差别主要在于 zone 设定中的类型不一样，主服务器为 master 类型，从服务器为 slave 类型。在对主 DNS 服务器端设定时要注意对正反解数据库文件的设置。

```
Oct 29 11:52:15 cgsp named[3596]: starting BIND 9.2.4 -u named -t /var/named/chroot
Oct 29 11:52:15 cgsp named[3596]: using 1 CPU
Oct 29 11:52:15 cgsp named: named startup succeeded
Oct 29 11:52:15 cgsp named[3596]: loading configuration from '/etc/named.conf'
Oct 29 11:52:15 cgsp named[3596]: listening on IPv4 interface lo, 127.0.0.1#53
Oct 29 11:52:15 cgsp named[3596]: listening on IPv4 interface eth0, 10.1.0.1#53
Oct 29 11:52:15 cgsp named[3596]: listening on IPv4 interface eth1, 211.65.63.146#53
Oct 29 11:52:15 cgsp named[3596]: listening on IPv4 interface ib1, 192.168.10.1#53
Oct 29 11:52:15 cgsp named[3596]: command channel listening on 127.0.0.1#953
Oct 29 11:52:15 cgsp named[3596]: command channel listening on ::1#953
Oct 29 11:52:15 cgsp named[3596]: running
Oct 29 11:52:15 cgsp named[3596]: zone 12.18.172.in-addr.arpa/IN: transferred serial 2007101813
Oct 29 11:52:15 cgsp named[3596]: transfer of '12.18.172.in-addr.arpa/IN' from 172.18.12.179#53: end of transfer
Oct 29 11:52:15 cgsp named[3596]: zone 12.18.172.in-addr.arpa/IN: sending notifies (serial 2007101813)
Oct 29 11:52:15 cgsp named[3596]: zone seu.edu.cn/IN: transferred serial 2007101801
Oct 29 11:52:15 cgsp named[3596]: transfer of 'seu.edu.cn/IN' from 172.18.12.179#53: end of transfer
Oct 29 11:52:15 cgsp named[3596]: zone seu.edu.cn/IN: sending notifies (serial 2007101801)
```

图 7-26　传送失败的日志文件

7.4　DNS 客户端的配置

应用实例导航——A 公司 DNS 客户端的配置与测试

※场景呈现

A 公司架设好主从 DNS 服务器后，接着就要对客户端进行配置，客户端包括 Windows、Linux 两种服务器；并需要在 Linux 客户端对 172.18.12.179 这台 DNS 服务器进行正反解析测试。

试分别写出配置 Windows 和 Linux 客户端的步骤，并利用 Linux 提供的 host、nslookup 和 dig 命令对 DNS 服务器端进行测试。

※技术要领

(1) Windows 客户端的配置。
(2) Linux 客户端的配置。
(3) host、nslookup 和 dig 命令的使用。

完成 DNS 服务器端，包括 Cache-only、DNS 主服务器和从服务器的配置以后，本节介绍 Linux 和 Windows 客户端的 DNS 配置，以及 Linux 系统下几个常用的测试 DNS 服务器的命令。

7.4.1　Linux 客户端的 DNS 配置

本节介绍 Linux 客户端的 DNS 配置步骤。

❶ DNS 客户端的配置涉及三个配置文件，分别为/etc/hosts、/etc/resolv.conf 和/etc/nsswitch.conf。/etc/hosts 文件已经提及多次，它记录了 IP 地址与主机域名的对应关系，提供了与 DNS 数据库同样的功能；/etc/resolv.conf 记录了 DNS 服务器的地址，即发送 DNS 查询请求的地址；/etc/nsswitch.conf 文件设定了是先查询/etc/hosts 文件，还是先使用/etc/resolv.conf 提供的 DNS

服务器地址发送请求，通常设为先/etc/hosts 后/etc/resolv.conf，在/etc/nsswitch.conf 中的设定如图 7-27 所示。

```
[root@seugrid2 ~]# vi /etc/nsswitch.conf

hosts:      files dns

#
```

图 7-27　/etc/nsswitch.conf 主机查询设定

❷ 将客户端所知道的 DNS 服务器地址写入/etc/resolv.conf 文件中，如图 7-28 所示。本例仅写了一个，即上面配置过的 172.18.12.179，一般写入 2～3 个 DNS 服务器地址比较合适，这样既防止单个服务器的不可用，又不浪费查询时间。

```
[root@seugrid2 ~]# vi /etc/resolv.conf
nameserver 172.18.12.179
~
~
```

图 7-28　/etc/resolv.conf 设定

7.4.2　Windows 客户端的 DNS 配置

本节介绍 Windows 客户端的 DNS 配置步骤。

❶ 在 Windows 下配置 DNS 客户端的方法也很简单，右击【网上邻居】图标，在弹出的快捷菜单中选择【属性】命令，弹出【网络连接】窗口。右击【本地连接】图标，在弹出的快捷菜单中选择【属性】命令，弹出【本地连接 属性】窗口，如图 7-29 所示。

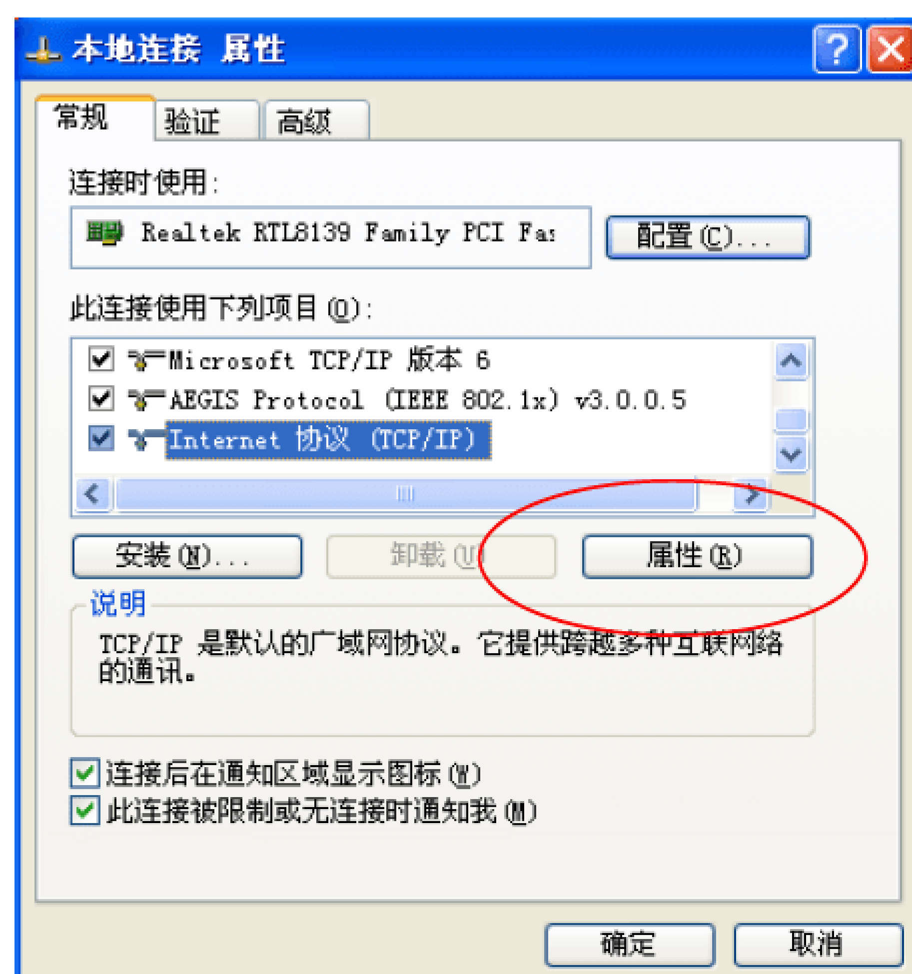


图 7-29　设置本地连接属性

❷ 在【本地连接 属性】窗口中选中【Internet 协议(TCP/IP)】复选框，然后单击【属性】按钮，如图 7-29 标注部分所示。此时系统会打开【Internet 协议(TCP/IP)属性】对话框，如图 7-30 所示。选中【使用下面的 IP 地址】单选按钮，然后在【首选 DNS 服务器】和【备用 DNS 服务器】文本框中输入 DNS 服务器的 IP 地址，单击【确定】按钮即可完成 Windows 下的 DNS 客户端的配置。

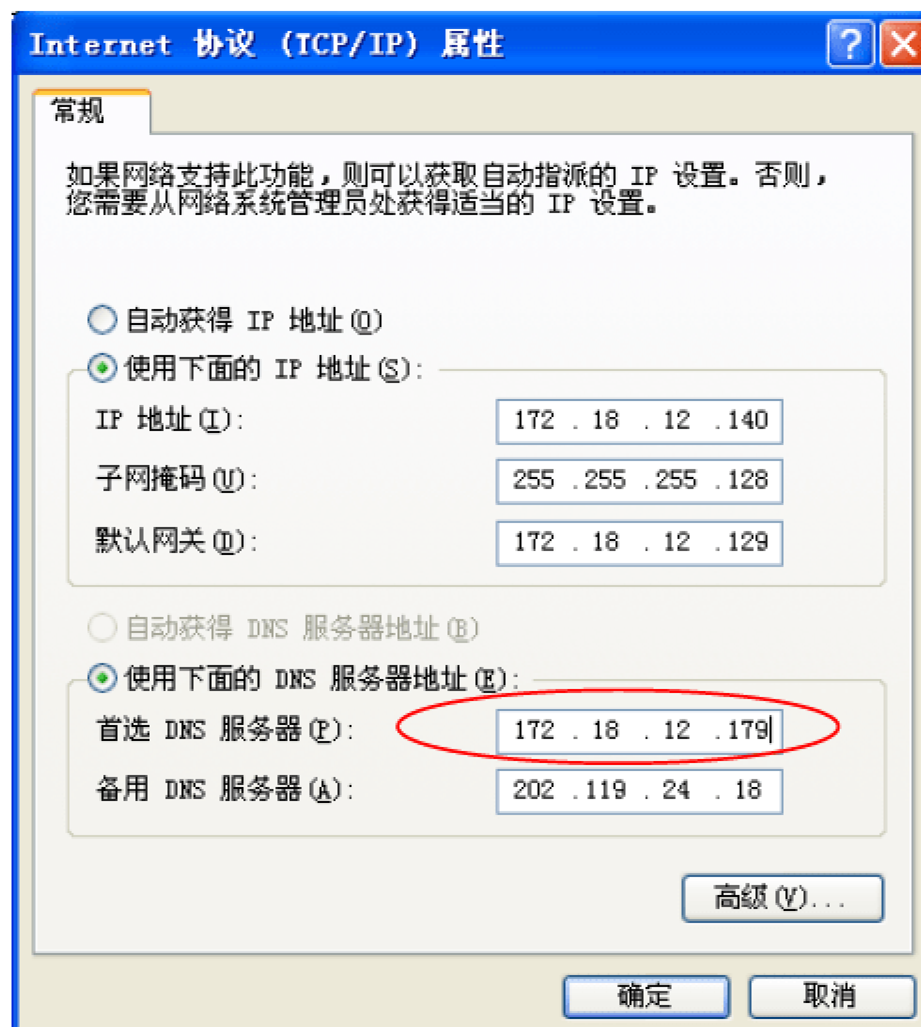


图 7-30　Internet 协议属性

7.4.3　DNS 客户端的测试命令

完成 DNS 客户端的简单配置后，本节介绍几个常用的 DNS 服务器测试命令，分别是 host、nslookup 和 dig 命令。

首先介绍 host 命令，该命令提供了查询 DNS 服务器信息和提交域名查询 IP 或提交 IP 查询域名等功能，使用下面命令来查询 DNS 服务器信息，结果如图 7-31 所示。

```
host -a 172.18.12.179
```

-a 参数表示列出 DNS 服务器的所有信息，包括正解反解信息、TTL、IP 信息等。

host 查询命令的格式为：

```
host [需要查询的域名或 IP] [DNS 服务器地址]
```

如图 7-32 所示，从 DNS 服务器 172.18.12.179 上查询域名为 seugrid2.seu.edu.cn 这台主机的 IP 地址。图 7-32 的标注部分显示该请求发送到了 172.18.12.179 DNS 服务器的 53 端口，返回结果为 seugrid2.seu.edu.cn，这台主机的 IP 地址是 172.18.12.178。图 7-33 显示了查询 seugrid1.seu.edu.cn 这台主机的命令和返回结果，这两台主机的 IP 对应关系都是我们在正向解析数据库文件中设置过的，返回结果与其一致。

```
[root@seugrid2 ~]# host -a 172.18.12.179
Trying "179.12.18.172.in-addr.arpa"
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 2821
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 1

;; QUESTION SECTION:
;179.12.18.172.in-addr.arpa.    IN      PTR

;; ANSWER SECTION:
179.12.18.172.in-addr.arpa. 600 IN      PTR     seugrid3.seu.edu.cn.

;; AUTHORITY SECTION:
12.18.172.in-addr.arpa. 600     IN      NS      cgsp.job.
12.18.172.in-addr.arpa. 600     IN      NS      seugrid3.seu.edu.cn.

;; ADDITIONAL SECTION:
seugrid3.seu.edu.cn.    600     IN      A       172.18.12.179

Received 129 bytes from 172.18.12.179#53 in 2 ms
```

图 7-31　host –a 命令

```
[root@seugrid2 ~]# host seugrid2.seu.edu.cn 172.18.12.179
Using domain server:
Name: 172.18.12.179
Address: 172.18.12.179#53
Aliases:

seugrid2.seu.edu.cn has address 172.18.12.178
```

图 7-32　查询 seugrid2.seu.edu.cn

```
[root@seugrid2 ~]# host seugrid1.seu.edu.cn 172.18.12.179
Using domain server:
Name: 172.18.12.179
Address: 172.18.12.179#53
Aliases:

seugrid1.seu.edu.cn has address 172.18.12.181
```

图 7-33　查询 seugrid1.seu.edu.cn

其次，介绍 nslookup 命令，nslookup 命令是 DNS 服务器的主要测试工具，它提供了执行 DNS 服务器查询测试并获取详细信息。使用 nslookup 命令可以诊断和解决名称解析问题、检查资源记录是否在区域中正确添加或更新以及排除其他服务器的其他问题。nslookup 有两种运行模式：非交互式和交互式。

nslookup 命令在非交互式正向查询的命令格式为：

```
nslookup [需查询的域名] [DNS 服务器地址]
```

正向查询例子如图 7-34 所示。非交互式反向查询的命令格式为：

```
nslookup [需查询的 IP] [DNS 服务器地址]
```

反向查询例子如图 7-35 所示。

在这个例子中，直接输入 nslookup 命令则进入交互模式，出现“>”提示符，比如我们想查询 www.sina.com 的 IP 地址，可以直接输入 www.sina.com，如图 7-36 所示，nslookup 连接到 172.18.12.179 这台 DNS 服务器上查询，并返回查询结果。

```
[root@seugrid2 ~]# nslookup seugrid1.seu.edu.cn 172.18.12.179
Server:          172.18.12.179
Address:         172.18.12.179#53

Name:    seugrid1.seu.edu.cn
Address: 172.18.12.181
```

图 7-34　nslookup 命令正向查询

```
[root@seugrid2 ~]# nslookup 172.18.12.181 172.18.12.179
Server:          172.18.12.179
Address:         172.18.12.179#53

181.12.18.172.in-addr.arpa      name = seugrid1.seu.edu.cn.
```

图 7-35　nslookup 命令反向查询

```
[root@seugrid2 ~]# nslookup
> www.sina.com
;; Truncated, retrying in TCP mode.
Server:          172.18.12.179
Address:         172.18.12.179#53

Non-authoritative answer:
www.sina.com    canonical name = us.sina.com.cn.
us.sina.com.cn  canonical name = news.sina.com.cn.
news.sina.com.cn        canonical name = jupiter.sina.com.cn.
jupiter.sina.com.cn     canonical name = hydra.sina.com.cn.
Name:   hydra.sina.com.cn
Address: 218.30.108.74
Name:   hydra.sina.com.cn
Address: 218.30.108.55
```

图 7-36　nslookup 命令交互模式

最后介绍 dig 命令，dig 命令提供了更为强大的功能，除了给出所要查询的结果外，还会附带查出域内其他的相关信息。如图 7-37 所示，利用 dig 命令查询 seugrid1.seu.edu.cn 的 IP，ANSWER SECTION 部分给出了精确的查询结果，AUTHORITY SECTION 部分给出了 seu.edu.cn 域内的其他主机名，ADDITIONAL SECTION 中还给出了正解数据库文件中设置的其他主机名和 IP 的对应关系。

点评与拓展： 本节只是介绍了 host、nslookup 和 dig 命令的基本用法，更多的扩展用法请读者在实践中体会。

```
[root@seugrid2 ~]# dig @172.18.12.179 seugrid1.seu.edu.cn

; <<>> DiG 9.3.2 <<>> @172.18.12.179 seugrid1.seu.edu.cn
; (1 server found)
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 61050
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 1

;; QUESTION SECTION:
;seugrid1.seu.edu.cn.           IN      A

;; ANSWER SECTION:
seugrid1.seu.edu.cn.    600     IN      A       172.18.12.181

;; AUTHORITY SECTION:
seu.edu.cn.             600     IN      NS      seugrid3.seu.edu.cn.
seu.edu.cn.             600     IN      NS      cgsp.job.

;; ADDITIONAL SECTION:
seugrid3.seu.edu.cn.    600     IN      A       172.18.12.179

;; Query time: 6 msec
;; SERVER: 172.18.12.179#53(172.18.12.179)
;; WHEN: Thu Oct 18 21:19:43 2007
;; MSG SIZE  rcvd: 114
```

图 7-37　dig 命令

7.5　本章小结

本章首先介绍了 DNS 的层次式域名结构、DNS 查询等基本原理；然后从两类最简单的 DNS 服务器 Cache-only 和 Forwarding 服务器出发，逐步深入地介绍根 DNS 服务器及其从服务器的配置；最后介绍 DNS 客户端的配置和测试。通过阅读本章，读者不仅需要了解 DNS 的基本原理和 Cache-only 和 Forwarding 服务器的特点和意义，更为重要的是学会配置根 DNS 服务器及其从服务器，并能够使用一些客户端命令测试所配置的 DNS 服务器是否能够正常工作。

第 8 章　Web 服务器的配置与架设

随着 Internet Web 服务的飞速发展，企业机构、政府部门、高等院校、科研院所等众多企事业单位都在积极构建自己的网站。由于 Web 服务是实现诸多网络应用的基础平台，配置和架设 Web 服务器是构建 Internet 和 Intranet 非常关键的工作。本章主要介绍如何使用 Apache 和 Tomcat 服务器软件来架设 Web 服务器，并介绍如何利用 Eclipse 开发基于 JSP 的 Web 工程及其在 Web 服务器上的部署工作。

通过本章的学习，读者应掌握以下基本内容：

- ✧ Web 服务的基本工作原理
- ✧ Apache 服务器的安装、配置和启动
- ✧ Tomcat 服务器的安装、配置和启动
- ✧ 使用 MyEclipse 开发 Web 工程
- ✧ Tomcat 服务器上 Web 工程的部署

8.1　Web 服务概述

Web 服务是互联网上最为重要的应用，是实现信息发布、资料查询、数据处理等诸多应用的基本平台，它使用 Hypertext(超级链接)方式将信息通过 Internet 传播到各地。

WWW(World Wide Web，万维网)的目的就是更容易获取信息，而不管信息的实际所在地理位置。当万维网使用超文本作为文档的标准格式后，人们开发了可以快速获取这些超文本文档的协议——HTTP 协议(即超文本传输协议)。

HTTP 协议是应用级协议，主要用于分布式协作的信息系统。HTTP 协议是通用无状态的，其系统建设和传输的数据无关。HTTP 协议也是面向对象的协议，可用于包括名字服务、分布式对象管理、请求方法的扩展、命令等各种任务。

在互联网上，HTTP 通信通常发生在 TCP/IP 连接上，其默认端口为 80，当然也可使用其他端口。

Web 服务采用 Client/Server(客户/服务器)模型来实现。客户端运行 Web 浏览器程序来浏览万维网，浏览器为用户提供了良好的用户界面。浏览器的作用是解释和显示 Web 页面，响应客户请求，并通过 HTTP 协议将客户请求传递给 Web 服务器。Web 服务器一端运行服务器程序，其基本功能是检测和响应客户端的 HTTP 请求，并向客户端发送处理结果。

Web 服务通常分为两种：静态 Web 服务和动态 Web 服务。所谓静态 Web 服务，就是服务器只是把存储的文档发送给客户端浏览器，此过程中传送的网页不会发生变化，除非网页制作人员利用制作工具对网页进行了修改。而动态 Web 服务则能够实现客户端浏览器

和服务器之间的交互。Web 服务器使用 CGI、ASP、PHP 和 JSP 等动态网站技术向浏览器发送动态内容。针对客户端浏览器发出的请求，Web 服务器在服务器端执行完响应程序，组织好文档后再将结果发回至客户端。

❶ Web 服务使用 HTTP(超文本传输协议)，该协议是一个 TCP/IP 协议基础上的应用级协议，其具体工作原理如图 8-1 所示。

Web 客户端浏览器使用 HTTP 命令向 Web 服务器发出页面请求。

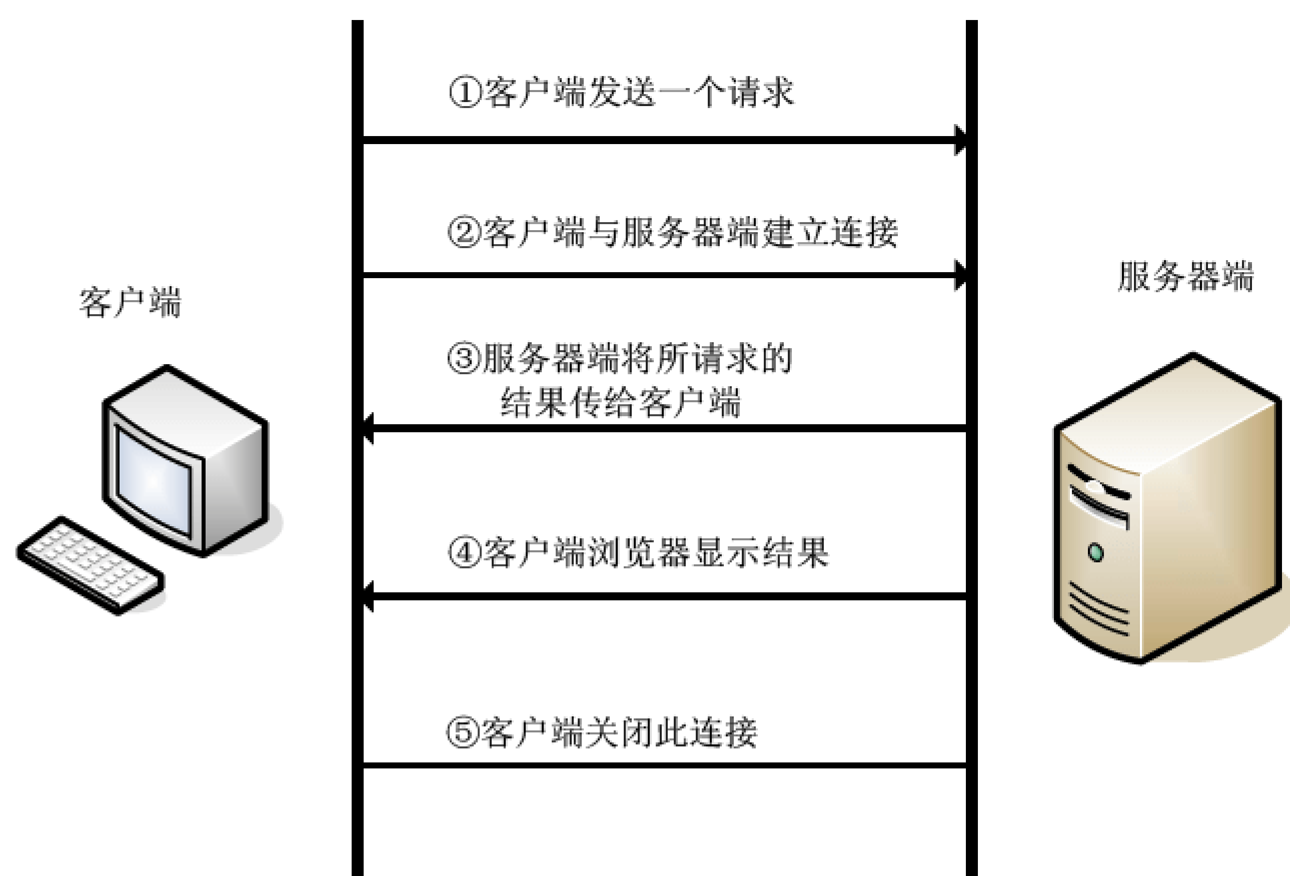


图 8-1　Web 服务工作原理

❷ 若服务器在特定端口(通常为 TCP80 端口)接收到页面请求，则发出一个应答，并在客户端和服务器之间建立连接。

❸ Web 服务器查找客户端所请求的文档。Web 服务器查找到所需文档后，就会将文档传给客户端浏览器。若该文档不存在，则 Web 服务器就会发送一个错误提示给客户端。

❹ 客户端浏览器接收到文档后，显示文档。

❺ 客户端浏览完毕后，断开与服务器的连接。

8.2　Apache 的安装、配置和启动

Apache(阿帕奇)服务器最初是由 Illinois 大学 Urbana-Champaign 的国家高级计算程序中心开发，后来被开放源代码团体的成员不断发展完善。起初，Apache 只是 Netscape 网页服务器之外的开放源代码选择。但它逐渐在速度和功能上超越其他 Web 服务器。20 世纪 90 年代中期以来，由于 Apache 服务器很强的可靠性，使其成为最流行的 Web 服务器。

据相关调查资料显示，截止 2005 年 10 月，Apache 的市场占有率约为 70%，同期 IIS 的市场占有率却只有 20%左右。由此可见 Apache 增长速度之迅速，其最大竞争者——微软 IIS 与其竟也有如此大的差距。

Apache 能发展如此迅速，是与其突出优点紧密相关的。首先，Apache 能在 UNIX、Linux 和 Windows 等诸多操作系统平台之上运行。其次，开放源代码的 Apache 得到全世界许多杰

出程序员的支持，他们为 Apache 编写了一系列强大的功能模块，使 Apache 能够无限扩展。并且，Apache 的工作性能和稳定性远远超越了其他同类产品。因此，广泛使用 Apache 作为 Web 服务器软件是非常必然的。

本节介绍 Apache 这种流行 Web 服务器的安装、配置与启动。

8.2.1 Apache 的安装

首先介绍如何安装 Apache 服务器，具体步骤如下。

1. Apache 是免费软件，可以到官方网站www.apache.org下载。图 8-2 显示了该网站的中国镜像站点的下载页面，上面有很多其他 Linux 系统软件安装包，而且下载速度较快，建议读者使用。将 apache_1.3.39.tar.gz 下载到 Windows 系统后，可以通过 SSH shell 所提供的 FTP 功能上传到服务器/usr/local 目录下，如图 8-3 所示。

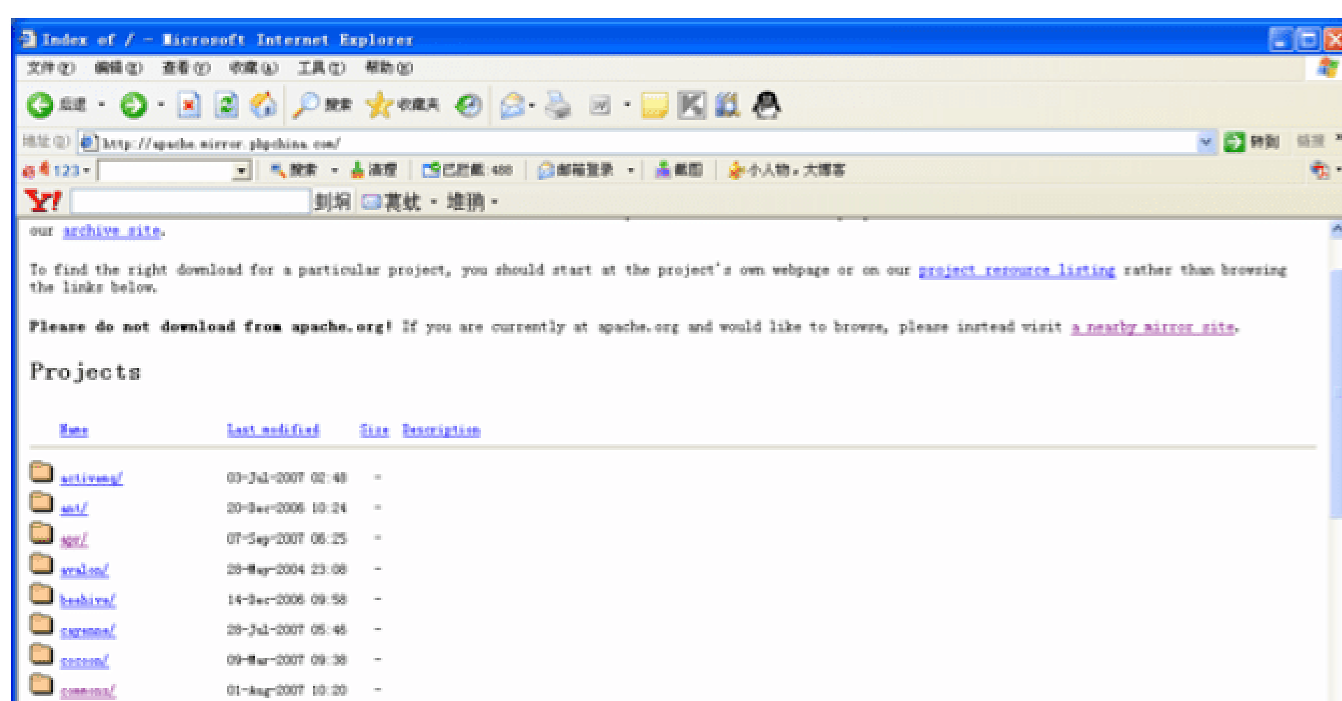


图 8-2 Apache 下载页面

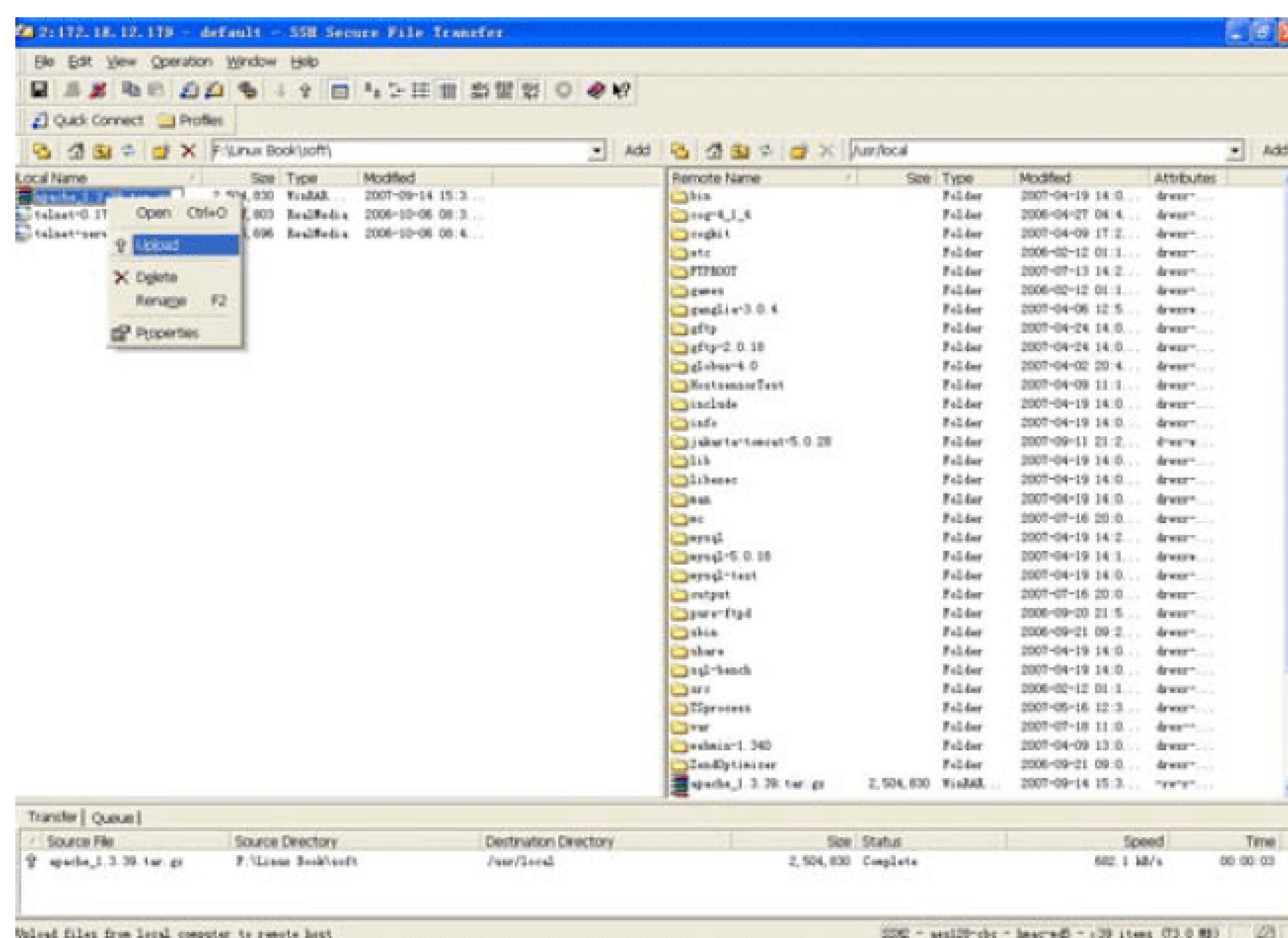


图 8-3 SSH 传输界面

❷ 将安装包上传到服务器后，首先使用下面命令将安装包解压缩，如图 8-4 所示。

```
tar -zxvf  apache_1.3.39.tar.gz
```

```
[root@seugrid3 local]# tar -zxvf apache_1.3.39.tar.gz
apache_1.3.39/
apache_1.3.39/ABOUT_APACHE
apache_1.3.39/cgi-bin/
apache_1.3.39/cgi-bin/printenv
apache_1.3.39/cgi-bin/test-cgi
apache_1.3.39/conf/
apache_1.3.39/conf/access.conf-dist
apache_1.3.39/conf/highperformance.conf-dist
apache_1.3.39/conf/httpd.conf-dist
apache_1.3.39/conf/httpd.conf-dist-nw
```

图 8-4　解压缩安装包

❸ Apache 这种类型的安装包，需要在 Linux 系统下手动进行配置、编译、安装。输入下面命令配置安装中的一些参数，通常需要指定安装目录，以命令参数--prefix 开头，运行结果如图 8-5 所示。

```
./configure --prefix=/usr/local/apache
```

```
[root@seugrid3 apache_1.3.39]# ./configure --prefix=/usr/local/apache
Configuring for Apache, Version 1.3.39
 + using installation path layout: Apache (config.layout)
Creating Makefile
Creating Configuration.apaci in src
Creating Makefile in src
 + configured for Linux platform
 + setting C compiler to gcc
 + setting C pre-processor to gcc -E
 + using "tr [a-z] [A-Z]" to uppercase
 + checking for system header files
 + adding selected modules
 + using system Expat
 + checking sizeof various data types
 + doing sanity check on compiler and options
Creating Makefile in src/support
Creating Makefile in src/regex
Creating Makefile in src/os/unix
Creating Makefile in src/ap
Creating Makefile in src/main
Creating Makefile in src/modules/standard
[root@seugrid3 apache_1.3.39]# 
```

图 8-5　配置 Apache 安装包

❹ 输入下面命令编译 Apache 安装包，运行结果如图 8-6 所示。

```
make
```

```
[root@seugrid3 apache_1.3.39]# make
===> src
make[1]: Entering directory `/usr/local/apache_1.3.39'
make[2]: Entering directory `/usr/local/apache_1.3.39/src'
===> src/regex
sh ./mkh  -i _REGEX_H_ regex2.h regcomp.c regerror.c regexec.c regfree.c > ../include/hsregex.h
gcc -I.  -I../os/unix -I../include   -DLINUX=22 -DHAVE_SET_DUMPABLE -DUSE_HSREGEX -DNO_DL_NEEDED `../apaci` -DPOSIX_MISTAKE   -c
gcc -I.  -I../os/unix -I../include   -DLINUX=22 -DHAVE_SET_DUMPABLE -DUSE_HSREGEX -DNO_DL_NEEDED `../apaci` -DPOSIX_MISTAKE   -c
gcc -I.  -I../os/unix -I../include   -DLINUX=22 -DHAVE_SET_DUMPABLE -DUSE_HSREGEX -DNO_DL_NEEDED `../apaci` -DPOSIX_MISTAKE   -c
gcc -I.  -I../os/unix -I../include   -DLINUX=22 -DHAVE_SET_DUMPABLE -DUSE_HSREGEX -DNO_DL_NEEDED `../apaci` -DPOSIX_MISTAKE   -c
rm -f libregex.a
```

图 8-6　编译 Apache 安装包

5 输入下面命令完成 Apache 的安装，运行结果如图 8-7 所示。

```
make install
```

```
[root@seugrid3 apache_1.3.39]# make install
make[1]: Entering directory `/usr/local/apache_1.3.39'
===> [mktree: Creating Apache installation tree]
./src/helpers/mkdir.sh /usr/local/apache/bin
mkdir /usr/local/apache
mkdir /usr/local/apache/bin
./src/helpers/mkdir.sh /usr/local/apache/bin
./src/helpers/mkdir.sh /usr/local/apache/libexec
```

图 8-7　安装 Apache 安装包

点评与拓展：Linux 系统下的解压过后的安装包下如果有 configure 这个可执行文件，一般都按照上述步骤安装。

8.2.2　Apache 的配置和启动

Apache 服务器最大的优点就是其配置的灵活性，所有 Apache 配置都使用一个文件，即 httpd.conf，该文件保存在目录/etc/httpd/conf 下，文件中的设置许多都只需使用默认值。我们在本节对 httpd.conf 的一些主要配置进行说明。

1 使用 vi 命令打开 Apache 的配置文件 httpd.conf 后，文件的第一部分是全局配置部分。经常需要修改的是 Timeout 和 MaxKeepAliveRequests 两个参数，如图 8-8 标注部分所示。Timeout 记录了每个操作的最大超时时间；MaxKeepAliveRequests 记录允许一个客户端连接同时能发出的最大请求数，如果设置为 0 表示没有限制，值越高服务器对客户端的服务性能就越高，但是服务器负载越大，建议根据服务器的承受能力选取合适的值。

```
#
# Timeout: The number of seconds before receives and sends time out.
#
Timeout 120

#
# KeepAlive: Whether or not to allow persistent connections (more than
# one request per connection). Set to "Off" to deactivate.
#
KeepAlive On

#
# MaxKeepAliveRequests: The maximum number of requests to allow
# during a persistent connection. Set to 0 to allow an unlimited amount.
# We recommend you leave this number high, for maximum performance.
#
MaxKeepAliveRequests 500
```

图 8-8　配置 Timeout 和 MaxKeepAliveRequests

全局配置的 MaxClients 记录了允许同时连接的客户端数量，如图 8-9 标注部分所示。如果发现在使用期间大量出现了拒绝客户端的连接，则可以尝试增加该数值；如果发现服务器在该期间锁定了或速度变慢，则可以尝试降低该数值。

```
# worker MPM
# StartServers: initial number of server processes to start
# MaxClients: maximum number of simultaneous client connections
# MinSpareThreads: minimum number of worker threads which are kept spare
# MaxSpareThreads: maximum number of worker threads which are kept spare
# ThreadsPerChild: constant number of worker threads in each server process
# MaxRequestsPerChild: maximum number of requests a server process serves
<IfModule worker.c>
StartServers         2
MaxClients         150
MinSpareThreads     25
MaxSpareThreads     75
ThreadsPerChild     25
MaxRequestsPerChild  0
</IfModule>
```

图 8-9　配置 MaxClients

❷ 接下来我们讲述配置主服务器部分，首先是设置 ServerAdmin 的 E-mail 地址，便于访问者通知 Web 站点所出现的问题，应将此处改为用户自己的 E-mail 地址，如图 8-10 所示。ServerName 设置主机名，这个设置很重要，如果设置错误，客户端无法通过主机名解析到 IP 地址，这个主机名必须是在/etc/hosts 配置文件中设置过的有效主机名，这样才能像 DNS 服务一样将主机名映射到 IP 地址。如图 8-11 所示/etc/hosts 中将主机名 seugrid3 与 IP 地址 172.18.12.179 绑定后，在 httpd.conf 中才可以将 ServerName 设为 seugrid3，如图 8-10 的标注部分所示。

```
# ServerAdmin: Your address, where problems with the server should be
# e-mailed.  This address appears on some server-generated pages, such
# as error documents.  e.g. admin@your-domain.com
#
ServerAdmin zawuster@gmail.com

#
# ServerName gives the name and port that the server uses to identify itself.
# This can often be determined automatically, but we recommend you specify
# it explicitly to prevent problems during startup.
#
# If this is not set to valid DNS name for your host, server-generated
# redirections will not work.  See also the UseCanonicalName directive.
#
# If your host doesn't have a registered DNS name, enter its IP address here.
# You will have to access it by its address anyway, and this will make
# redirections work in a sensible way.
#
ServerName seugrid3
```

图 8-10　配置 ServerAdmin 和 ServerName

```
[root@seugrid3 html]# vi /etc/hosts
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1       localhost.localdomain   localhost
172.18.12.179   seugrid3.seu.edu.cn     seugrid3
172.18.12.181   seugrid1.seu.edu.cn     seugrid1
172.18.12.178   seugrid2.seu.edu.cn     seugrid2
~
```

图 8-11　/etc/hosts 配置文件

❸ Apache 的主服务器端配置部分的另一个重要配置就是配置存放网页 HTML 等文档的目录，

它记录在参数 DocumentRoot 中，我们设置为/var/www/html，也就是说，我们只有把所有的 HTML 文件放入此目录，才能找到 Apache，如图 8-12 所示。

```
#
# DocumentRoot: The directory out of which you will serve your
# documents. By default, all requests are taken from this directory, but
# symbolic links and aliases may be used to point to other locations.
#
DocumentRoot "/var/www/html"
```

图 8-12　配置 DocumentRoot

❹ DirectoryIndex 记录了 DocumentRoot 中的首页名称和顺序，用户需要将网站首页的 HTML 的名字填入此处，如图 8-13 所示。当客户端试图连接 Apache 的站点时，Apache 进行检查是否有定义的 DirectoryIndex，它按顺序逐一检查每个已命名的文件，优先选择排在前面的文件，图 8-13 中所示优先选择 index.html。如果 index.html 和 index.html.var 都不存在，将返回 403 “禁用”错误或一个目录索引。

```
# DirectoryIndex: sets the file that Apache will serve if a directory
# is requested.
#
# The index.html.var file (a type-map) is used to deliver content-
# negotiated documents.  The MultiViews Option can be used for the
# same purpose, but it is much slower.
#
DirectoryIndex index.html index.html.var
```

图 8-13　配置 DirectoryIndex

❺ 使用下面命令可以启动、停止、重启动 Apache 服务器，图 8-14 演示了这个命令的几种使用格式。其中，/etc/rc.d/init.d/httpd status 是查看 Apache 服务器的状态，正常运行的 Apache 服务器启动了 8 个进程；/etc/rc.d/init.d/httpd reload 是重读配置文件，重新载入 httpd 服务。

```
/etc/rc.d/init.d/httpd {stop|start|restart|reload|status|help....}
```

```
[root@seugrid3 ~]# /etc/rc.d/init.d/httpd status
httpd (pid 17457 17456 17455 17454 17453 17452 17451 17450 23514) 正在运行...
[root@seugrid3 ~]# /etc/rc.d/init.d/httpd stop
停止 httpd:                                                [确定]
[root@seugrid3 ~]# /etc/rc.d/init.d/httpd start
启动 httpd:                                                [确定]
[root@seugrid3 ~]# /etc/rc.d/init.d/httpd reload
重新载入 httpd:                                            [确定]
```

图 8-14　Apache 服务器的启动与停止

❻ 为了测试 Apache 服务器，我们写了个简单的 HTML 文件，index.html，如图 8-15 所示，并将其部署到 Apache 服务器，HTML 的详细规范请参考有关资料。编辑完 index.html 后，将其复制到前面设置的 DocumentRoot 目录中。打开浏览器，输入http://localhost，Apache 自动定位到 index.html，显示结果如图 8-16 所示。

```
[root@seugrid3 html]# vi index.html
<html>

<head>
<meta http-equiv="Content-Type" content="text/html; charset=GBK" >



<title>Test Aapche Index</title>
<!--mstheme--><link rel="stylesheet" href="image/netw1011-106.css">
<meta name="Microsoft Theme" content="network 1011">
</head>

<body   >
<p>  </p>
<p>  </p>
<p align="center"><b><blink><font size="7" color="red">It is Apache FrontPage</font></blink></b></p>
<!--<%=new java.util.Date|).toString() %>-->
<p>  </p>
<p>  </p>
<p>  </p>
<p>  </p>
<p align="center"><b><font size="5">Welcome you!</font></a></b></p>


</body>

</html>
```

图 8-15　index.html 文件

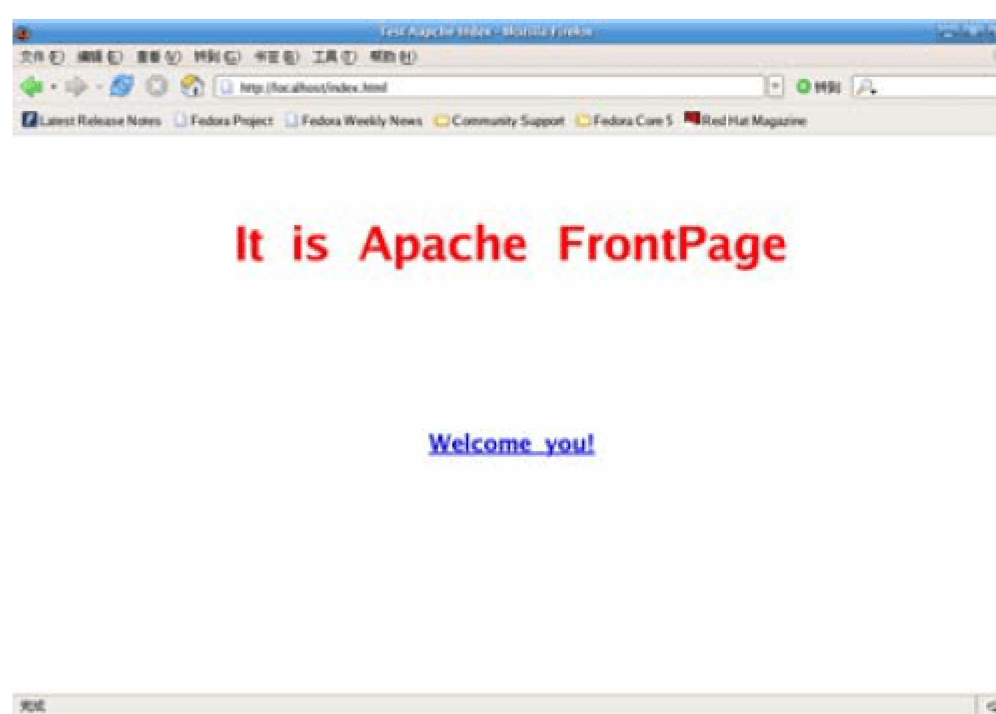


图 8-16　测试 Apache 服务器

📖**点评与拓展**：Apache 服务器启动命令与前面几章讲述的一些服务的启动命令略有不同，细微之处，不可不察。

8.3　Tomcat 的安装和启动

应用实例导航——A 公司 Tomcat 服务器架设

※场景呈现

A 公司需要架设 Tomcat 服务器，以提供对动态 JSP 网页的支持；这个 Tomcat 服务器

可以通过 Web 的形式来管理它。

要求安装和配置 Tomcat 服务器以满足以上要求，假设需要将 Tomcat 管理员账号设定为 admin，密码也为 admin。

※技术要领

(1) J2DK 的安装。
(2) Tomcat 的安装、配置与启动。
(3) Tomcat 管理用户的配置。

Tomcat 是基于 Java 的，运行 Servlet 和 JSP Web 应用软件的开放源代码 Web 应用软件容器，它主要是由 Apache、Sun 及其他一些公司共同开发而成，并且是 Apache 基金会 Jakarta 项目中的一个主要项目。Sun 的参与和支持，使得 Tomcat 在相当程度上体现了最新的 Servlet 和 JSP 规范。由于 Tomcat 是根据 Servlet 和 JSP 规范执行的，所以可以说 Tomcat 也实行了 Apache-Jakarta 规范，且比许多同类应用软件要好。

尽管 Tomcat 可以独立作为 Web 服务器运行，但 Tomcat 的 Web 功能远没有 Apache 强大，所以在实际应用中是通过 mod_jk2 连接器(connetors)将 Apache 和 Tomcat 整合在一起提供服务。

与 Apache 相比，Tomcat 不仅支持 HTML 静态网页，而且还提供对 JSP 动态网页的支持。由于 Tomcat 功能强大、运行效率高、安装管理方便，因此它已经成为了目前架设 Web 服务器的首选软件。本节介绍 Tomcat 的安装、启动及其相关配置。

8.3.1　J2DK 的安装

在安装 Tomcat 前，必须先安装 Sun 公司的 J2DK，即 Java 2 Development Kit(Java 开发工具包)。它是一种用于构建 Java 平台上发布的应用程序、Applet 和组件的开发环境，是一切 Java 应用程序的基础，版本号为 1_5_0 以上的 J2DK 已经可以满足当前 Java 应用程序的基本需要，本节以 1_5_0_04 版本为例介绍其安装过程。安装包可以到官方网站 http://Java.sun.com/j2se/1.5.0/download.jsp下载，或者在本书附带光盘中找到。安装包的名字为：jdk-1_5_0_04-linux-i586.bin。

❶ 将安装包复制到 Linux 下的目录，一般为/usr 目录，然后为该文件添加运行权限，使用下面命令：

```
chmod 700 jdk-1_5_0_04-linux-i586.bin
```

然后使用下面命令运行该安装包：

```
./ jdk-1_5_0_04-linux-i586.bin
```

如图 8-17 所示，输入安装包的运行命令后，安装程序显示 Sun Mrcrosystems 的协议，按 Enter 键阅读。

❷ 阅读完协议后，安装程序询问用户是否接受该协议，输入 yes 接受，如图 8-18 所示。

```
[root@seugrid3 usr]# chmod 700 jdk-1_5_0_04-linux-i586.bin
[root@seugrid3 usr]# ./jdk-1_5_0_04-linux-i586.bin
Sun Microsystems, Inc. Binary Code License Agreement

for the JAVA 2 PLATFORM STANDARD EDITION DEVELOPMENT KIT 5.0

SUN MICROSYSTEMS, INC. ("SUN") IS WILLING TO LICENSE THE SOFTWARE
IDENTIFIED BELOW TO YOU ONLY UPON THE CONDITION THAT YOU ACCEPT
ALL OF THE TERMS CONTAINED IN THIS BINARY CODE LICENSE AGREEMENT AND
SUPPLEMENTAL LICENSE TERMS (COLLECTIVELY "AGREEMENT"). PLEASE READ THE
AGREEMENT CAREFULLY. BY DOWNLOADING OR INSTALLING THIS SOFTWARE, YOU
ACCEPT THE TERMS OF THE AGREEMENT. INDICATE ACCEPTANCE BY SELECTING THE
"ACCEPT" BUTTON AT THE BOTTOM OF THE AGREEMENT. IF YOU ARE NOT WILLING
TO BE BOUND BY ALL THE TERMS, SELECT THE "DECLINE" BUTTON AT THE BOTTOM
OF THE AGREEMENT AND THE DOWNLOAD OR INSTALL PROCESS WILL NOT CONTINUE.

1. DEFINITIONS. "Software" means the identified above in binary form,
any other machine readable materials (including, but not limited to,
libraries, source files, header files, and data files), any updates or
error corrections provided by Sun, and any user manuals, programming
guides and other documentation provided to you by Sun under this
Agreement. "Programs" mean Java applets and applications intended to
run on the Java 2 Platform Standard Edition (J2SE platform) platform
on Java-enabled general purpose desktop computers and servers.

2. LICENSE TO USE. Subject to the terms and conditions of this
Agreement, including, but not limited to the Java Technology
Restrictions of the Supplemental License Terms, Sun grants you a
non-exclusive, non-transferable, limited license without license fees
to reproduce and use internally Software complete and unmodified
for the sole purpose of running Programs. Additional licenses for
developers and/or publishers are granted in the Supplemental License
Terms.

3. RESTRICTIONS. Software is confidential and copyrighted. Title to
Software and all associated intellectual property rights is retained
by Sun and/or its licensors. Unless enforcement is prohibited by
```

图 8-17 J2DK 安装命令

```
For inquiries please contact: Sun Microsystems, Inc., 4150 Network
Circle, Santa  Clara, California 95054, U.S.A.  (LFI#141623/Form
ID#011801)

Do you agree to the above license terms? [yes or no]
```

图 8-18 接受协议

❸ 接受协议后，安装程序自动解压并安装其中的 RPM 安装包，如图 8-19 所示。安装完毕后，/usr 目录下出现了 jdk1_5_0_04 文件夹，这就是 J2DK 的安装目录。

```
Creating jdk1.5.0_04/lib/tools.jar
Creating jdk1.5.0_04/jre/lib/rt.jar
Creating jdk1.5.0_04/jre/lib/jsse.jar
Creating jdk1.5.0_04/jre/lib/charsets.jar
Creating jdk1.5.0_04/jre/lib/ext/localedata.jar
Creating jdk1.5.0_04/jre/lib/plugin.jar
Creating jdk1.5.0_04/jre/lib/javaws.jar
Creating jdk1.5.0_04/jre/lib/deploy.jar

Done.
[root@seugrid3 usr]# ls
apache-ant-1.6.2              bin  games    jdk1.5.0_04                  kerberos  libexec  sbin   src  X11R6
apache-ant-1.6.2-bin.tar.gz   etc  include  jdk-1_5_0_04-linux-i586.bin  lib       local    share  tmp
[root@seugrid3 usr]#
```

图 8-19 J2DK 的安装目录

📖**点评与拓展**：Linux 系统下的.bin 格式的安装包都类似于 J2DK 的安装方法，先将安装包的权限设置为可执行，然后再运行该安装包。

8.3.2　Tomcat 的安装和配置

安装好 J2DK 后，就可以开始安装 Tomcat 了，较新的 Tomcat 版本是 5.0 以上的版本，安装包可以到官方网站http://jakarta.apache.org/site/downloads/downloads_tomcat-5.cgi 下载，或者在本书附带光盘中找到。安装包的名字为 jakarta-tomcat-5.0.28.tar.gz。本节就以 5.0.28 版本的 Tomcat 为例介绍其安装过程，详细步骤如下。

❶ 将安装包复制到 Linux 下的目录，一般为/usr/local 目录，由于 Tomcat 安装包是 rar 压缩包，所以直接解压该安装包即可，命令如下：

```
tar -zxvf jakarta-tomcat-5.0.28.tar.gz
```

解压后出现/usr/local/jakarta-tomcat-5.0.28 目录，图 8-20 显示了 Tomcat 的安装目录结构。

```
[root@seugrid3 jakarta-tomcat-5.0.28]# ls
bin  common  conf  LICENSE  logs  NOTICE  RELEASE-NOTES  RUNNING.txt  server  shared  temp  webapps  work
[root@seugrid3 jakarta-tomcat-5.0.28]#
```

图 8-20　Tomcat 的安装目录结构

❷ 安装好 Tomcat 后，还需要设置两个环境变量才能正常启动 Tomcat，一个是 JAVA_HOME，它指向 Java 的安装目录，利于其他应用程序找到 J2DK；另一个是 CATALINA_HOME，它指向 Tomcat 安装目录。可以用下面命令设置：

```
export JAVA_HOME=/usr/jdk1.5.0_04
export CATALINA_HOME=/usr/local/ jakarta-tomcat-5.0.28
```

有经验的网络管理员通常会将所有设置环境变量的命令放在用户目录下的.bash_profile 文件中，比如，root 用户就放在/root/.bash_profile 下，这样每次重新启动服务器后只需执行.bash_profile 这个文件就能完成所有环境变量的设置工作，而不需要重复地输入 export 命令。我们用 vi 命令打开.bash_profile 文件，将上面的两条语句写入其中，如图 8-21 所示。第一次执行.bash_profile 前千万不要忘了添加执行权限，然后再运行，两条命令如下：

```
chmod 700 .bash_profile
. .bash_profile
```

如图 8-22 所示。请注意：执行.bash_profile 的命令中两个点之间有空格。

我们输入 set 命令就可以看见刚才设置过的两个环境变量了，set 命令回显所有该用户设定的环境变量，如图 8-22 中标注部分所示。

图 8-21 中其他几个环境变量也需要读者注意，比如，CLASSPATH，这个环境变量记录了 Java 程序寻找类库的默认路径，如果 Java 类不包含在$JAVA_HOME/lib 目录下，这些类库将无法被找到，这时就需要将包含这些类库的目录添加到 CLASSPATH 中。PATH 变量记录了系统寻找可执行文件的默认路径，PATH 目录、/bin 以及/usr/sbin 目录下所有的可执行文件可以直接被执行，而无需进入相应的根目录。

```
# .bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
        . ~/.bashrc
fi

# User specific environment and startup programs
export GLOBUS_LOCATION=/usr/local/globus-4.0
export JAVA_HOME=/usr/jdk1.5.0_04
export COG_HOME=/usr/local/cogkit/cog/dist/cog-4_1_4
export CLASSPATH=/usr/jdk1.5.0_04/lib
export CATALINA_HOME=/usr/local/jakarta-tomcat-5.0.28
PATH=$JAVA_HOME/bin:$COG_HOME/bin:$PATH:$HOME/bin
#/home/httpd/etherusb.cfg
export PATH
unset USERNAME
~
```

图 8-21 .bash_profile 命令

```
[root@seugrid3 ~]# chmod 700 .bash_profile
[root@seugrid3 ~]# . .bash_profile
[root@seugrid3 ~]# set
BASH=/bin/bash
BASH_ARGC=()
BASH_ARGV=()
BASH_LINENO=()
BASH_SOURCE=()
BASH_VERSINFO=([0]="3" [1]="1" [2]="7" [3]="1" [4]="release" [5]="i386-redhat-linux-gnu")
BASH_VERSION='3.1.7(1)-release'
CATALINA_HOME=/usr/local/jakarta-tomcat-5.0.28
CLASSPATH=/usr/jdk1.5.0_04/lib
COG_HOME=/usr/local/cogkit/cog/dist/cog-4_1_4
COLORS=/etc/DIR_COLORS
COLUMNS=160
CVS_RSH=ssh
DIRSTACK=()
EUID=0
GLOBUS_LOCATION=/usr/local/globus-4.0
GROUPS=()
G_BROKEN_FILENAMES=1
HISTFILE=/root/.bash_history
HISTFILESIZE=1000
HISTSIZE=1000
HOME=/root
HOSTNAME=seugrid3
HOSTTYPE=i386
IFS=$' \t\n'
INPUTRC=/etc/inputrc
JAVA_HOME=/usr/jdk1.5.0_04
KDEDIR=/usr
LANG=zh_CN.GB18030
```

图 8-22 执行.bash_profile 后的结果

❸ 设置好环境变量后，就可以直接使用环境变量进入 Tomcat 安装目录，输入下面命令。其中 conf 目录存放了 Tomcat 的相关配置文件，其中最重要的一个是 server.xml。首先演示如何更改 Tomcat 服务器的端口号，利用 vi 命令打开 server.xml，如图 8-23 所示，两条命令分列于下:

```
cd $CATALINA_HOME
vi conf/server.xml
```

```
[root@seugrid3 ~]# cd $CATALINA_HOME
[root@seugrid3 jakarta-tomcat-5.0.28]# ls
bin  common  conf  LICENSE  logs  NOTICE  RELEASE-NOTES  RUNNING.txt  server  shared  temp  webapps  work
[root@seugrid3 jakarta-tomcat-5.0.28]# cd conf
[root@seugrid3 conf]# ls
Catalina  catalina.policy  catalina.properties  jk2.properties  server-minimal.xml  server.xml  tomcat-users.xml  web.xml
[root@seugrid3 conf]# vi server.xml
<!-- Example Server Configuration File -->
<!-- Note that component elements are nested corresponding to their
     parent-child relationships with each other -->
```

图 8-23　打开 server.xml 文件

❹ 打开 server.xml 文件后，找到如图 8-24 所示的部分，Tomcat 的默认端口为 8080，现在将其改成 9080，如图标注部分。保存 server.xml 文件就完成了 Tomcat 的端口配置。

```
    <!-- Define a non-SSL Coyote HTTP/1.1 Connector on port 8080 -->
    <Connector port="9080"
               maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
               enableLookups="false" redirectPort="8443" acceptCount="100"
               debug="0" connectionTimeout="20000"
               disableUploadTimeout="true" />
    <!-- Note : To disable connection timeouts, set connectionTimeout value
     to 0 -->
```

图 8-24　Tomcat 端口配置

📖**点评与拓展**：.bash_profile 是 Linux 系统为每个系统用户自动新建的隐藏文件，在系统用户的根目录下。每个系统用户都需要设置自己的环境变量，可以在切换到不同用户后立即执行.bash_ profile 文件完成环境变量的设置。

8.3.3　Tomcat 的启动

安装并配置好 Tomcat 后，就可以启动并测试 Tomcat 服务器，具体步骤如下。

❶ 在 Tomcat 安装目录下只需输入图 8-25 所示的命令就能启动，如果在其他目录下，可以输入下面完整命令启动：

```
.$CATALINA_HOME/bin/startup.sh
```

```
[root@seugrid3 jakarta-tomcat-5.0.28]# ./bin/startup.sh
Using CATALINA_BASE:   /usr/local/jakarta-tomcat-5.0.28
Using CATALINA_HOME:   /usr/local/jakarta-tomcat-5.0.28
Using CATALINA_TMPDIR: /usr/local/jakarta-tomcat-5.0.28/temp
Using JAVA_HOME:       /usr/jdk1.5.0_04
[root@seugrid3 jakarta-tomcat-5.0.28]#
```

图 8-25　启动 Tomcat

❷ 为了测试 Tomcat 是否启动成功，在任意主机的浏览器上输入http://IP:port，即会出现如图 8-26 所示的界面，也可在本机浏览器上输入http://localhost:9080进行测试。

❸ 输入下面的命令来关闭 Tomcat，如图 8-27 所示。

```
.$CATALINA_HOME/bin/shutdown.sh
```

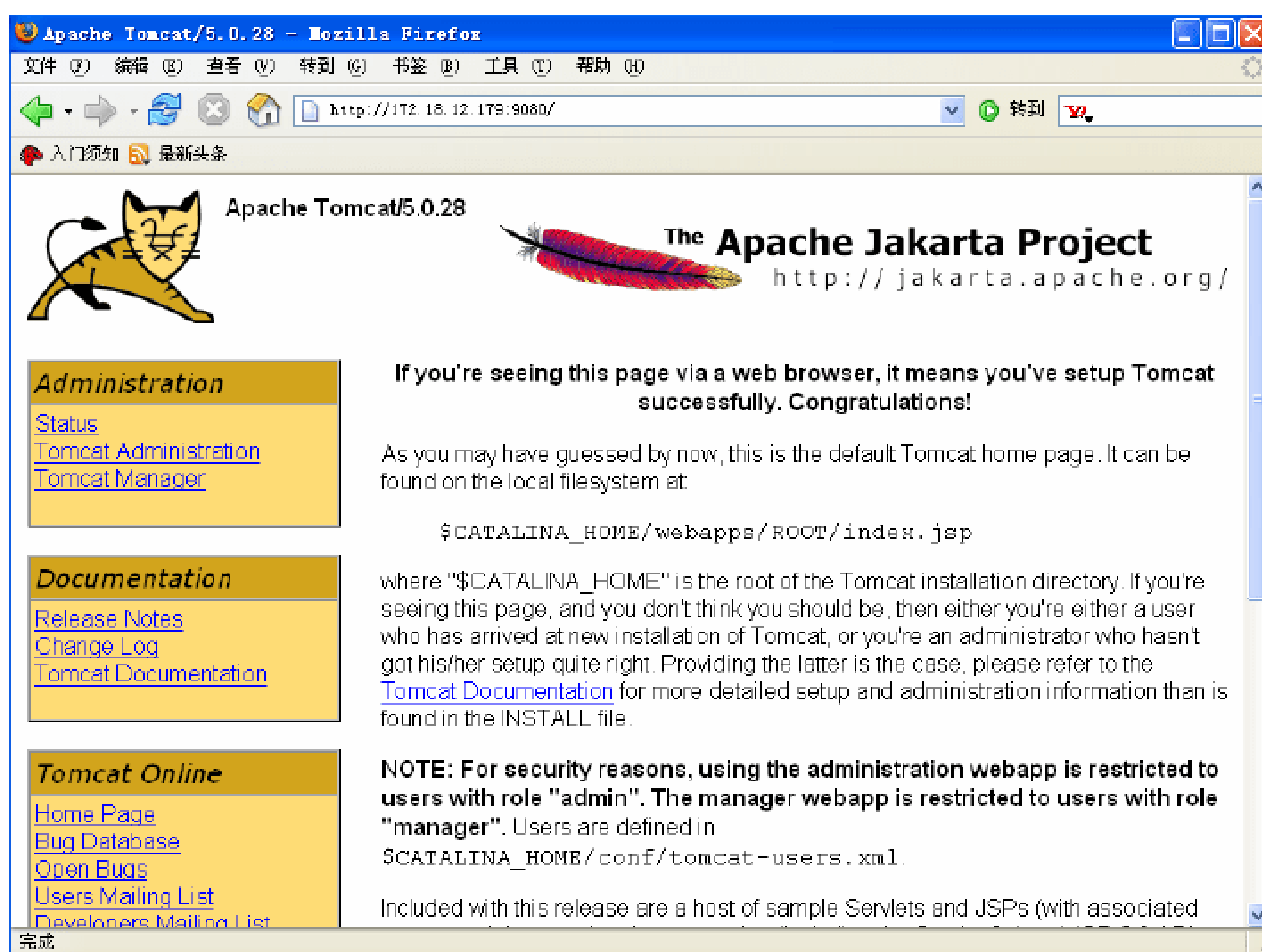


图 8-26 Tomcat 启动成功

```
[root@seugrid3 jakarta-tomcat-5.0.28]# ./bin/shutdown.sh
Using CATALINA_BASE:   /usr/local/jakarta-tomcat-5.0.28
Using CATALINA_HOME:   /usr/local/jakarta-tomcat-5.0.28
Using CATALINA_TMPDIR: /usr/local/jakarta-tomcat-5.0.28/temp
Using JAVA_HOME:       /usr/jdk1.5.0_04
[root@seugrid3 jakarta-tomcat-5.0.28]#
```

图 8-27 关闭 Tomcat

点评与拓展： 请注意，在设置环境变量的 export 命令后面没有“$”符号，但是使用环境变量进入目录时，需要加上“$” 符号，如 cd $JAVA_HOME。另外，当启动 Tomcat 而浏览器仍然不能访问 Tomcat 主页，这时应确认防火墙是否将 Tomcat 所占用的端口开启。

8.3.4 Tomcat 管理用户的配置

Tomcat 像其他 J2SE 环境一样，提供了方便的系统管理界面，这个 Web 形式的管理界面可以方便地管理 Tomcat 用户、Tomcat 数据库以及 Web 应用等。在使用这个系统管理界面前，我们需要先在 Tomcat 下进行相关的配置。当前网上大部分论述 Tomcat 管理用户配置的文档不但不全面，而且存在一些错误，因此本节提出了正确的配置方法。

❶ 前面也提及，$CATALINA_HOME 目录下的 conf 目录中存放了 Tomcat 的配置文件，其中 tomcat-users.xml 是专门针对 Tomcat 用户进行配置的文件。在添加 admin 权限的用户时，需要首先在这个文件下加入下面两行，如图 8-28 所示。

```
<role rolename="admin">
<user username="admin" password="secret" roles="admin"/>
```

我们所添加的 admin 用户是 Tomcat 的超级用户，它有权限添加其他的一切用户。

```
[root@seugrid3 jakarta-tomcat-5.0.28]# vi conf/tomcat-users.xml
<?xml version='1.0' encoding='utf-8'?>
<tomcat-users>
  <role rolename="tomcat"/>
  <role rolename="role1"/>
  <role rolename="manager"/>
  <role rolename="admin"/>
  <user username="tomcat" password="tomcat" roles="tomcat"/>
  <user username="mage" password="secret" roles="manager"/>
  <user username="both" password="tomcat" roles="tomcat,role1"/>
  <user username="role1" password="tomcat" roles="role1"/>
  <user username="admin" password="secret" roles="admin"/>
</tomcat-users>
~
```

图 8-28　tomcat-users.xml 配置文件

❷ 接下来是对$CATALINA_HOME/server/webapps 目录的配置，这个目录下有两个子目录，分别是 admin 和 manager，前者是用来设定 admin 用户的。进入 admin 目录，打开 admin.xml 这个配置文件，如图 8-29 所示。首先确保 docBase 参数的设置准确，它指向 admin 目录，设置为：

```
docBase=$CATALINA_HOME/server/webapps/admin
```

当然也可以像图 8-29 中那样设成绝对路径。

然后在下面加入与 tomcat-users.xml 一致的用户配置信息，如图 8-29 的标注部分。

```
<user name="admin" password="secret" roles="admin"/>
```

```
<Context path="/admin" docBase="/usr/local/jakarta-tomcat-5.0.28/server/webapps/admin"
        debug="0" privileged="true">

  <!-- Uncomment this Valve to limit access to the Admin app to localhost
   for obvious security reasons. Allow may be a comma-separated list of
   hosts (or even regular expressions).
  <Valve className="org.apache.catalina.valves.RemoteAddrValve"
    allow="127.0.0.1"/>
  -->

  <Logger className="org.apache.catalina.logger.FileLogger"
             prefix="localhost_admin_log." suffix=".txt"
          timestamp="true"/>
  <User name="admin"
       password="secret"
           roles="admin"/>
</Context>
```

图 8-29　admin.xml 配置文件

❸ 完成以上两步的设定后，在浏览器地址栏输入http://IP:port/admin就会弹出如图 8-30 所示的系统管理界面，提示输入用户名和密码。

❹ 输入上面设定的用户名和密码后(本书中是 admin 和 secret)，就进入如图 8-31 所示的系统管

理菜单，在上面可以添加其他用户或修改已存在的用户，这样配置 Tomcat 就无需使用 shell 命令行来完成，提高了网络管理员的效率。



图 8-30　Tomcat 管理界面登录

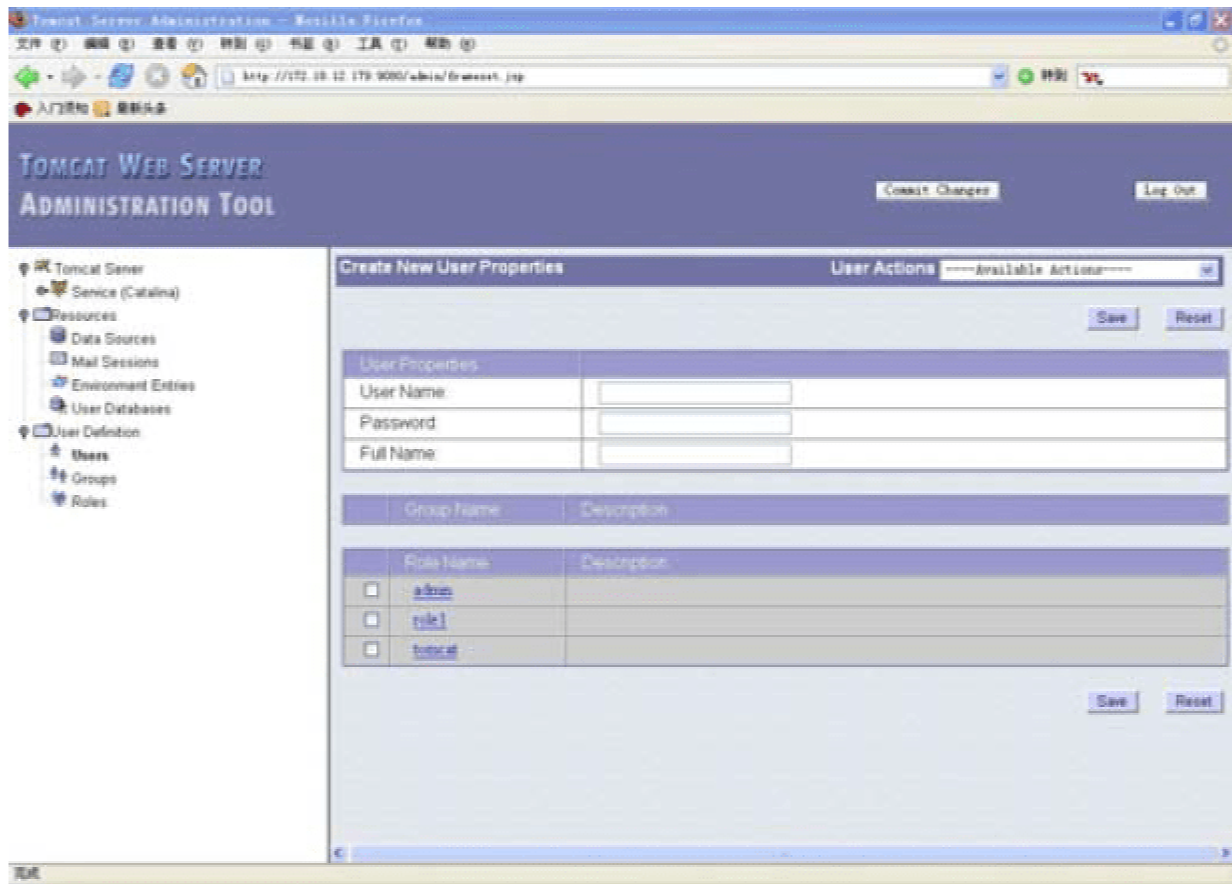


图 8-31　Tomcat 管理界面菜单

8.4　Web 工程的开发和部署

应用实例导航——在 Tomcat 下开发 Web 工程

※场景呈现

在为 A 公司架设好 Tomcat 服务器后，需要为 A 公司开发 JSP 动态网页，并部署到 Tomcat

服务器上。

要求示例性地开发一个 index.jsp 页面作为 A 公司 Tomcat 服务器的入口页面，然后在 index.jsp 页面上建立一个连接 showtime.jsp 的页面，这个页面可以显示当前的系统时间。

※技术要领

(1) MyEclipse 的使用。
(2) .war 文件的生成。
(3).war 文件部署到 Tomcat 服务器。

8.4.1 使用 MyEclipse 开发 JSP 网页

Eclipse 是一个开放源代码的、基于 Java 的可扩展开发平台。就其本身而言，它只是一个框架和一组服务，用于通过插件组件构建开发环境。幸运的是，Eclipse 附带了一个标准的插件集，包括 Java 开发工具(Java Development Tools，JDT)。Eclipse 还包括了插件开发环境(Plug-in Development Environment，PDE)，这个组件主要针对希望扩展 Eclipse 的软件开发人员，因为它允许他们构建与 Eclipse 环境无缝集成的工具。

MyEclipse 是 Eclipse 的插件，也是一款功能强大的 J2EE 集成开发环境，支持代码编写、配置、测试以及除错。Genuitec 发布了 MyEclipse Enterprise Workbench 5.0，标志着更智能、更快捷、更简单和更便宜的 J2EE 工具的新版本的诞生。它的价格对于个人和企业开发人员来说都是非常有吸引力的。通过增加 UML 双向建模工具、WYSIWYG 的 JSP/Strutsdesigner、可视化的 Hibernate/ORM 工具、Spring 和 Web services 支持，以及新的 Oracle 数据库开发，MyEclipse 5.0 继续为业界提供全面的产品。

MyEclipse 5.1 包括集成版本(All-in-One)的安装已经正式发布。MyEclipse 5.1 是基于 Eclipse 3.2.1 版本的，如果你的机器上已经安装了 Eclipse 3.2.1，只需要下载 Plugin 版本即可。如果你的 Eclipse 还没有更新到 3.2.1，MyEclipse All-in-OneInstaller 将为你安装更新到最新的 3.2.1，包括 Eclipse 3.2.1、MyEclipse 5.1 和 Java 5 JRE。注意：目前 All-in-One Installer 仅限于 Windows 操作系统用户使用，而 Mac 和 Linux 操作系统用户需要安装 Eclipse 3.2.1 并安装 MyEclipse 5.1。由于 Eclipse 3.2.1 是绿色软件，解压后即可以使用，MyEclipse 的安装也很简易，因此本书省略了 Eclipse 3.2.1 和 MyEclipse 的安装过程。(本书附带光盘提供了 Eclipse 3.2.1 和 MyEclipse 的安装包)

❶ 打开 Eclipse，选择 File→New→Project 命令创建一个新的 Java Web 工程，如图 8-32 标记部分所示，然后会弹出如图 8-33 所示的 New Project 窗口，选择 Web Project 类型，单击 Next 按钮进入下一步。

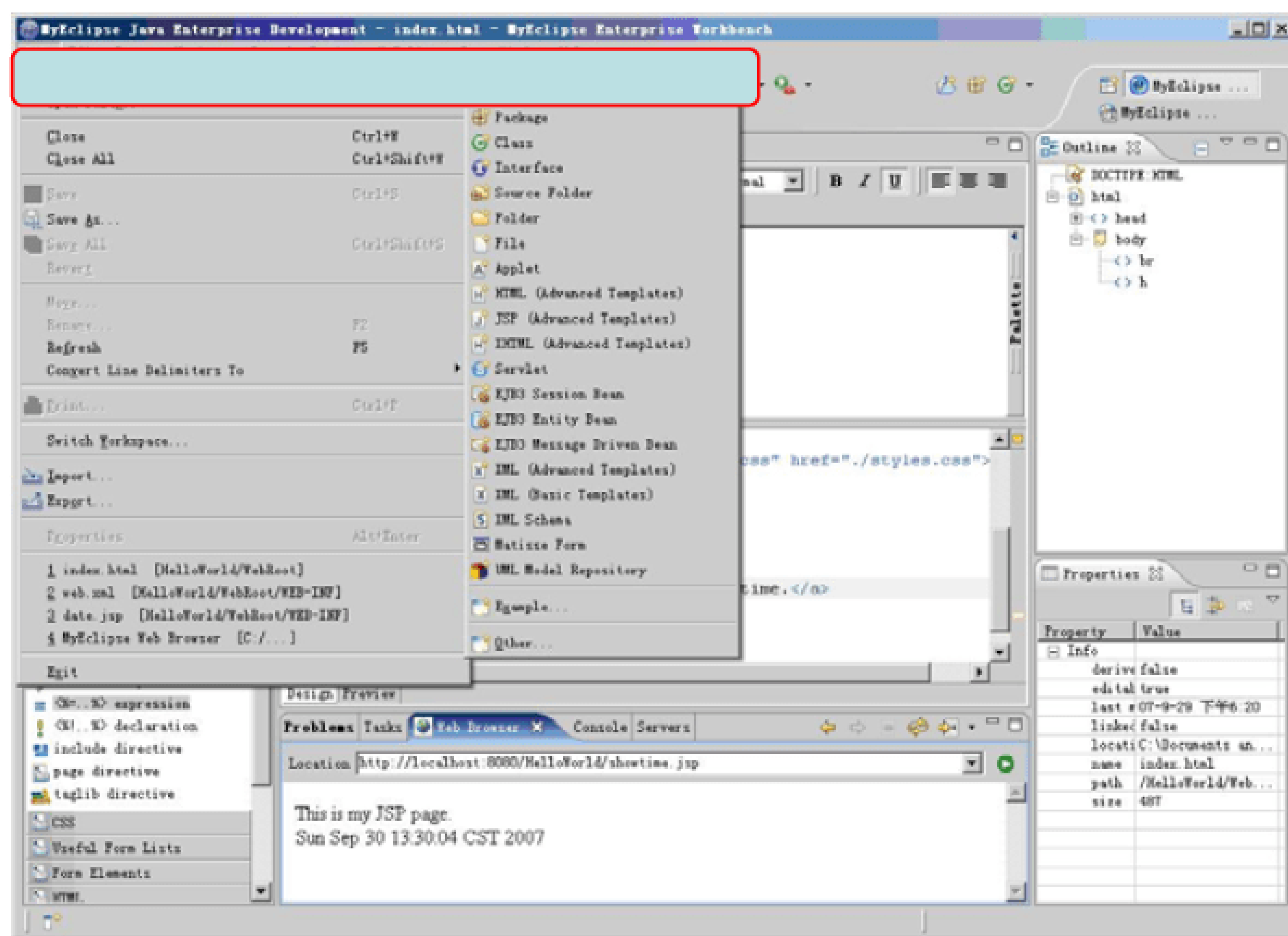


图 8-32　创建 Java Web 工程

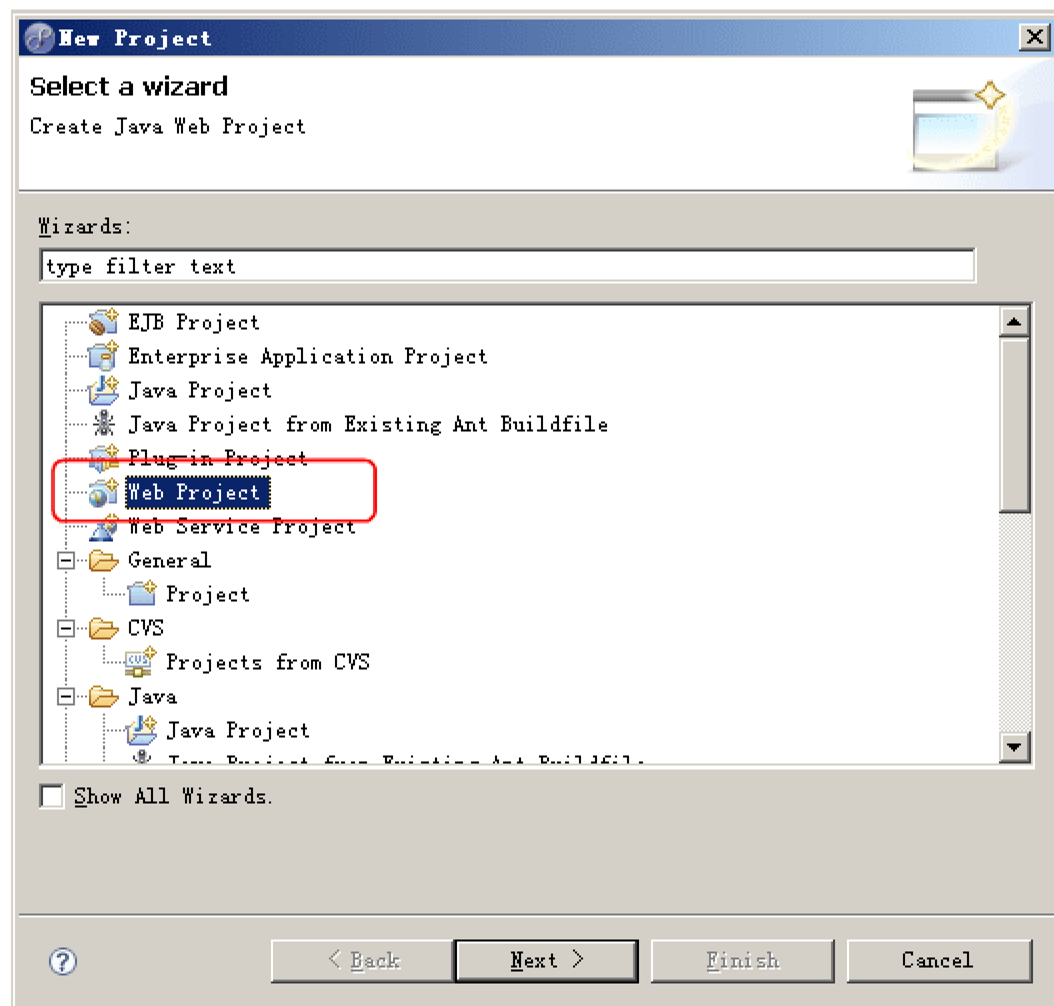


图 8-33　选择工程类型

❷ 接着弹出 New Web Project 对话框，供用户设置该 Web 工程的属性。在 Project Name 文本框中输入工程名称，本例输入的是 HelloWorld1，如图 8-34 所示，其他采用默认参数，单击 Finish 按钮完成该 Java Web 工程的创建。

❸ 工程名为 HelloWorld1 的 Java Web 工程创建成功后，出现 HelloWorld1→WebRoot 的目录结构，该目录下有个名字为 index.jsp 的文件，如图 8-35 所示，这个文件就是该工程的首页，当然也可将这个首页改成 index.html 文件。双击打开 index.jsp 文件，如图 8-36 所示，在图中标注部分可以添加你需要的 JSP 代码，由于本书讨论范围所限，对 JSP 规范不作详细介

绍，有兴趣的读者可以参考相关资料。

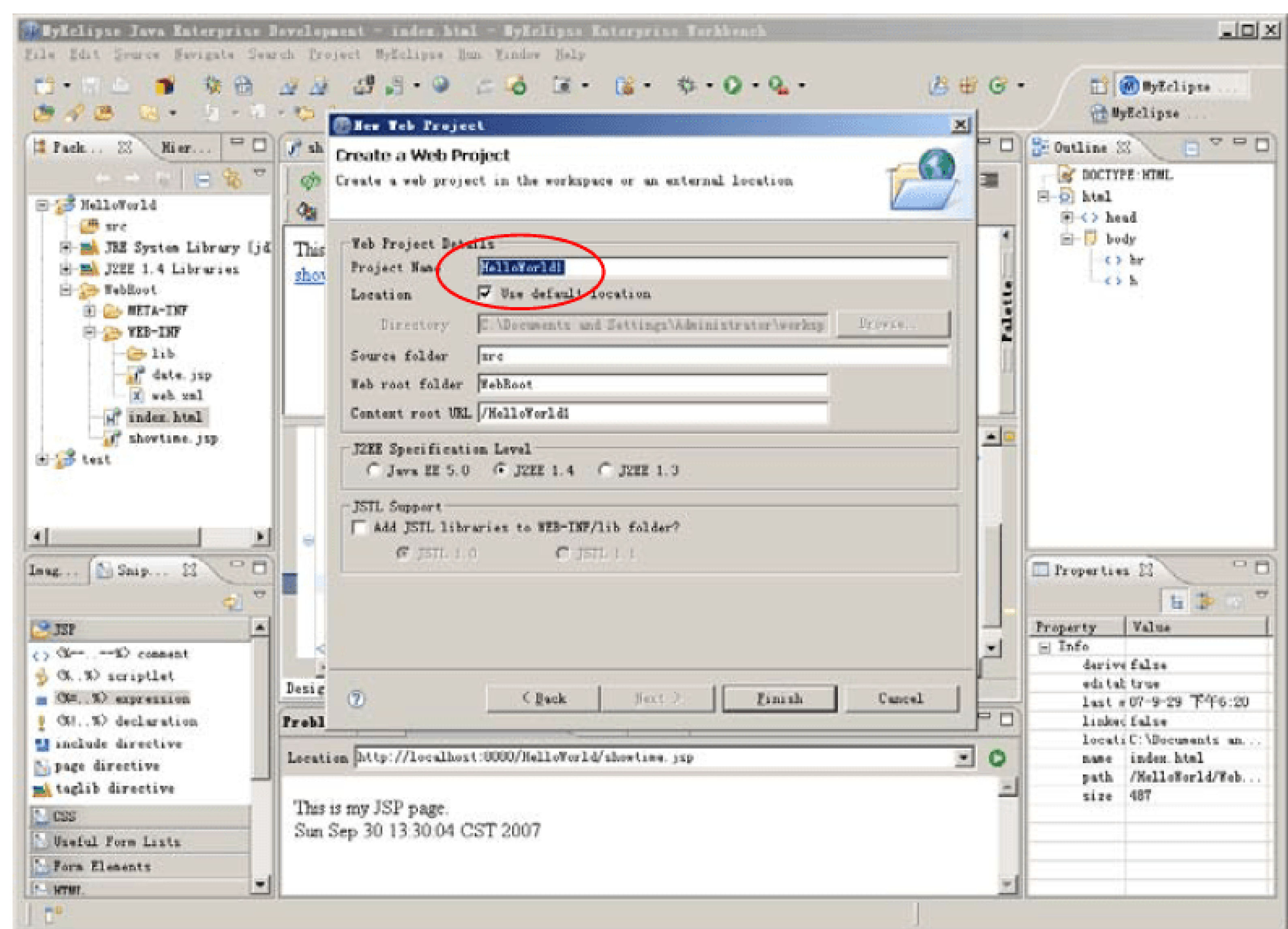


图 8-34　完成工程的创建

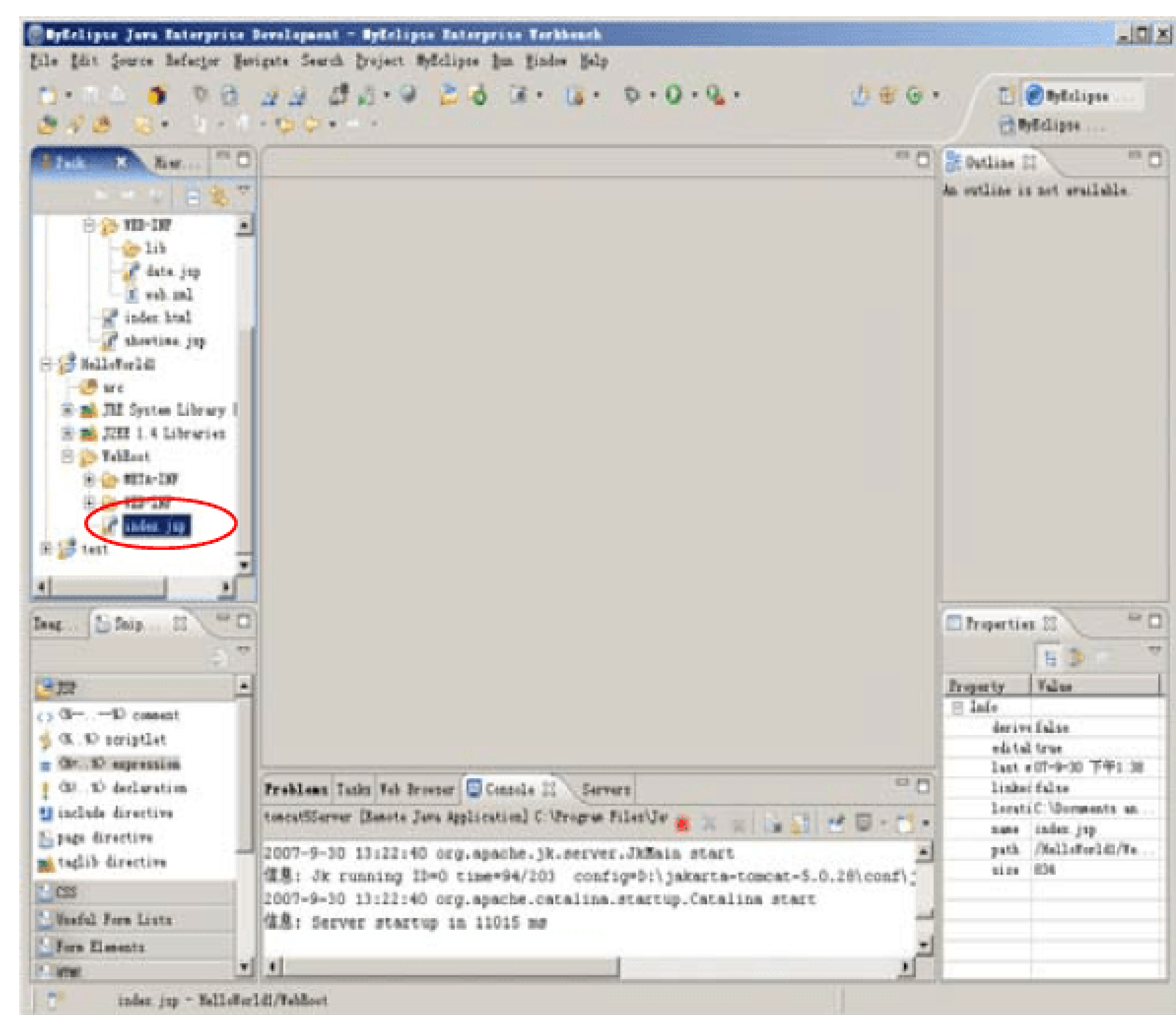


图 8-35　选择 index.jsp

❹ 接下来演示如何在 HelloWorld1 这个工程下创建新的 JSP 网页。单击工具栏第一个白色的图标，弹出创建新的文件类型列表，如图 8-37 所示。选择 JSP 类型，弹出如图 8-38 所示的对话框，指定该 JSP 文件的路径，一般存放在 WebRoot 目录下，并要求输入 JSP 文件名，本例中为 showtime.jsp，单击 Finish 按钮完成该 JSP 文件的创建。

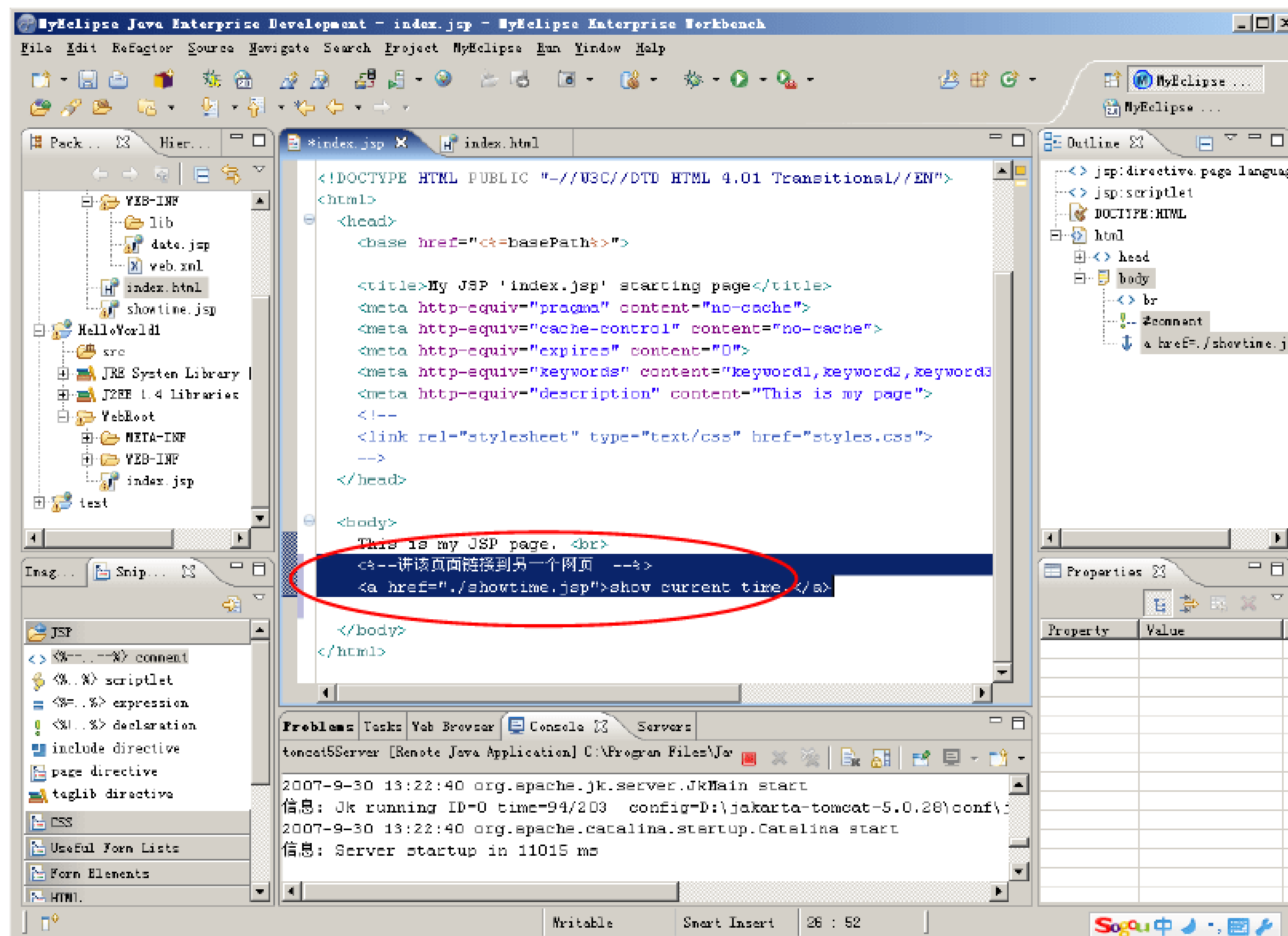


图 8-36　编辑 index.jsp

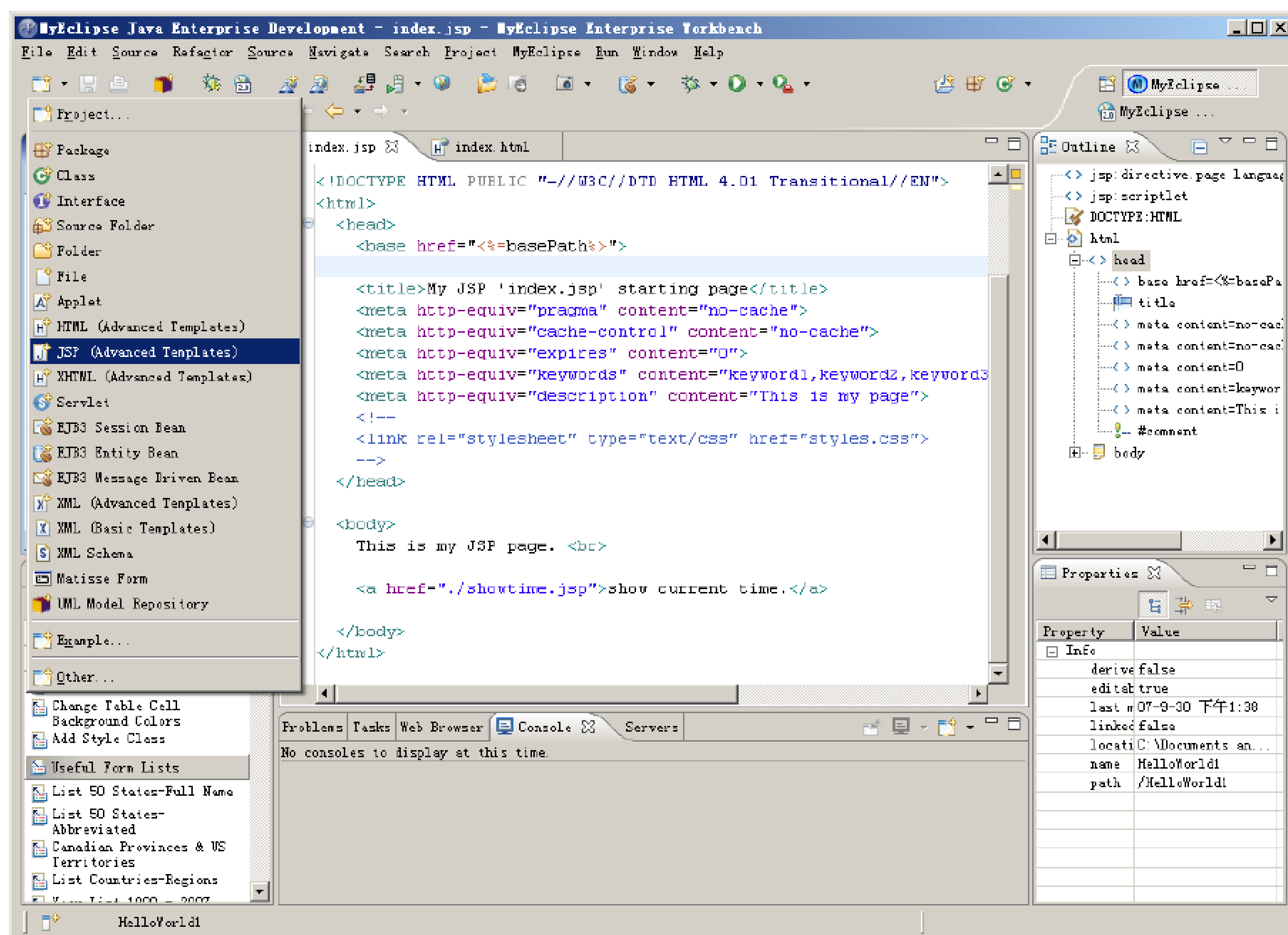


图 8-37　创建新的 JSP 页面

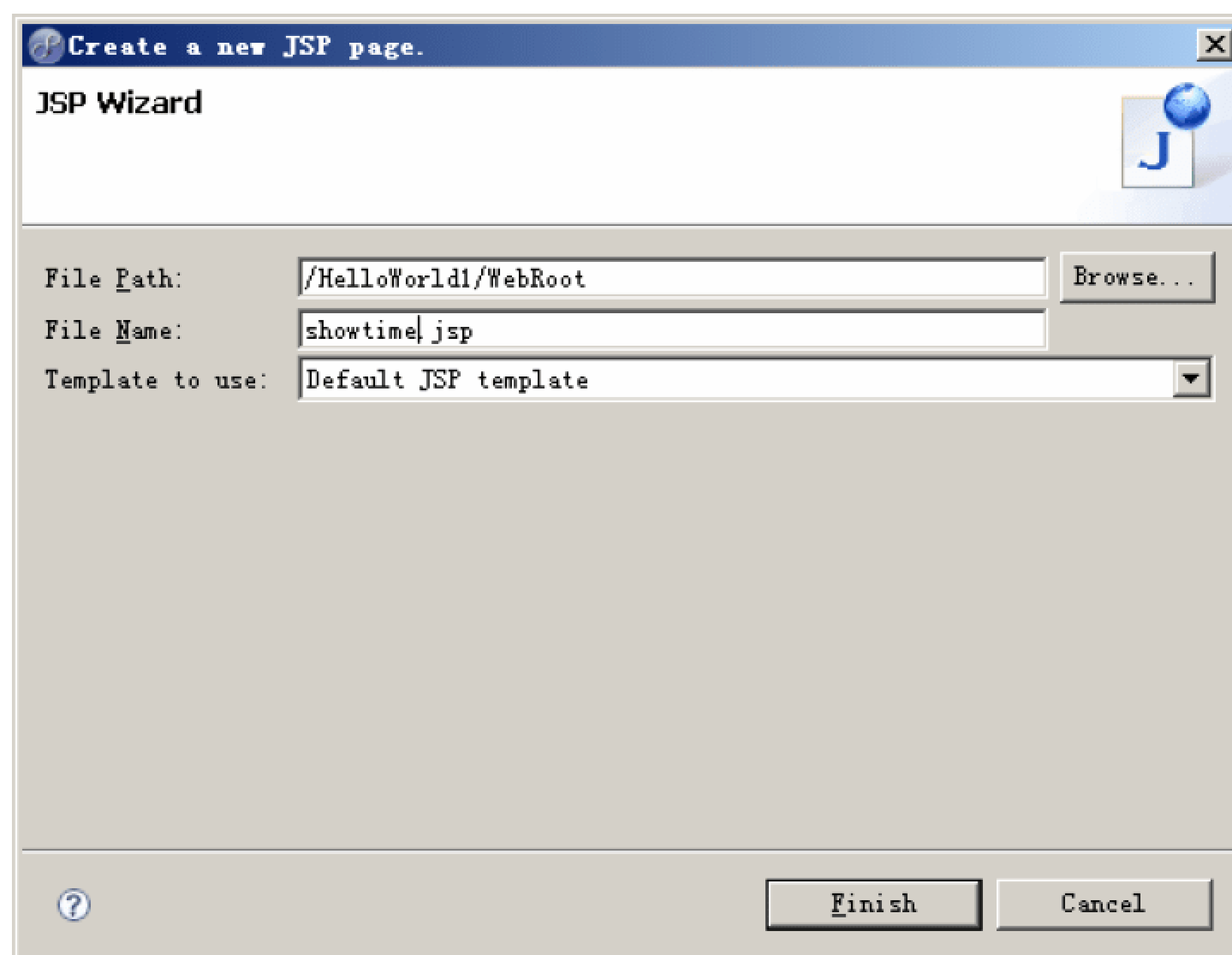


图 8-38　输入 JSP 文件名

❺ 创建成功 showtime.jsp 之后，可以双击打开，如图 8-39 所示。同样可以在标签<body>和</body>之间添加 JSP 代码，如图 8-39 标注部分所示。

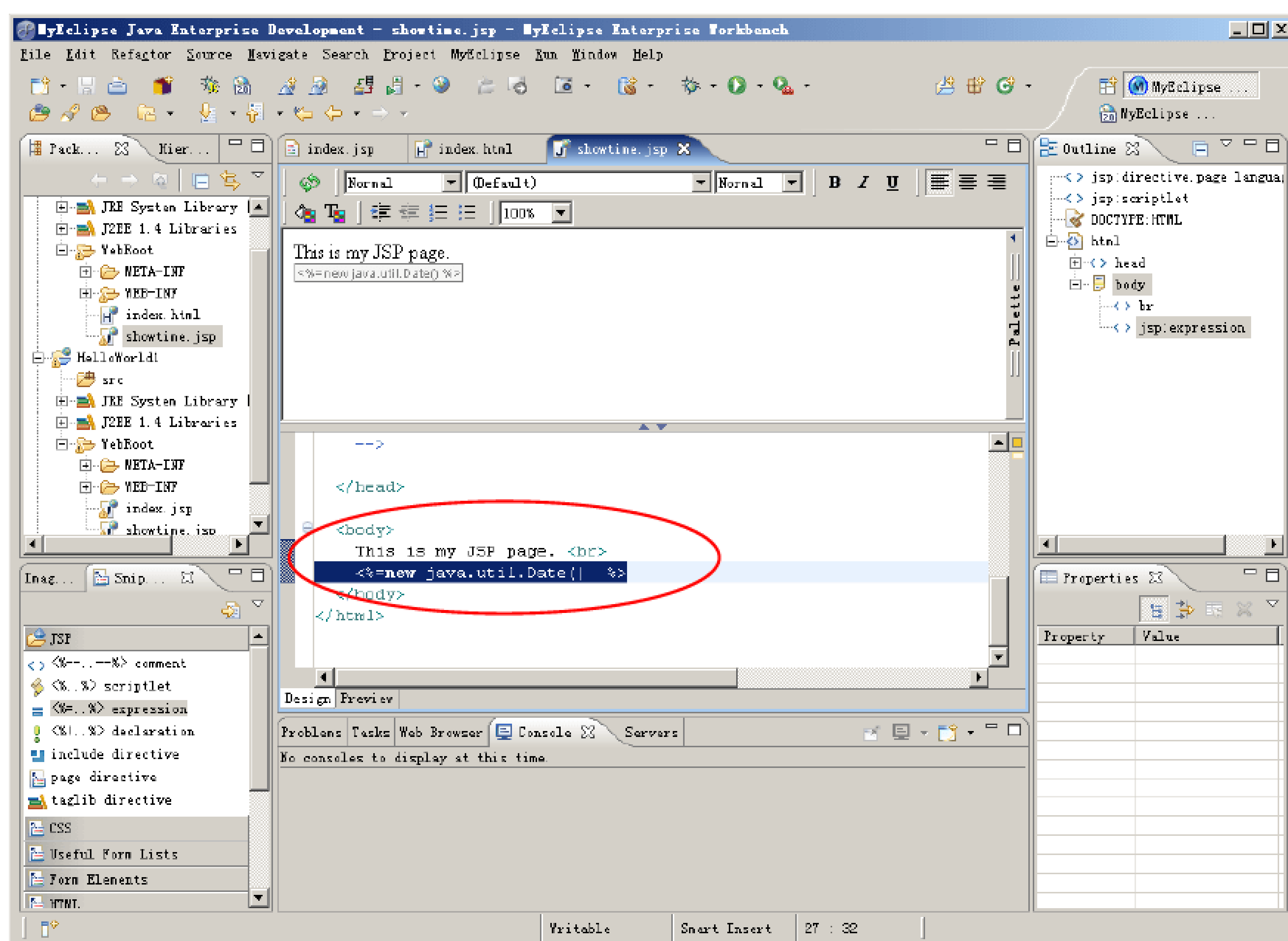


图 8-39　编辑 showtime.jsp

8.4.2 将 Web 工程发布成.war 文件

要将刚才所创建的 Web 工程部署到 Tomcat 服务器中，需要将应用程序包装为 Web 应用程序归档(Web Application Archive，WAR)文件。.war 文件类似于一个压缩包，整合了 Web 应用，它就像一个目录一样存在，使用相对路径或者目录名来使用。下面介绍如何将一个 Web 工程发布成.war 文件。

❶ 在 Eclipse 软件主窗口中选择 File→Export 命令发布上节所创建的 Web 工程，如图 8-40 所示。

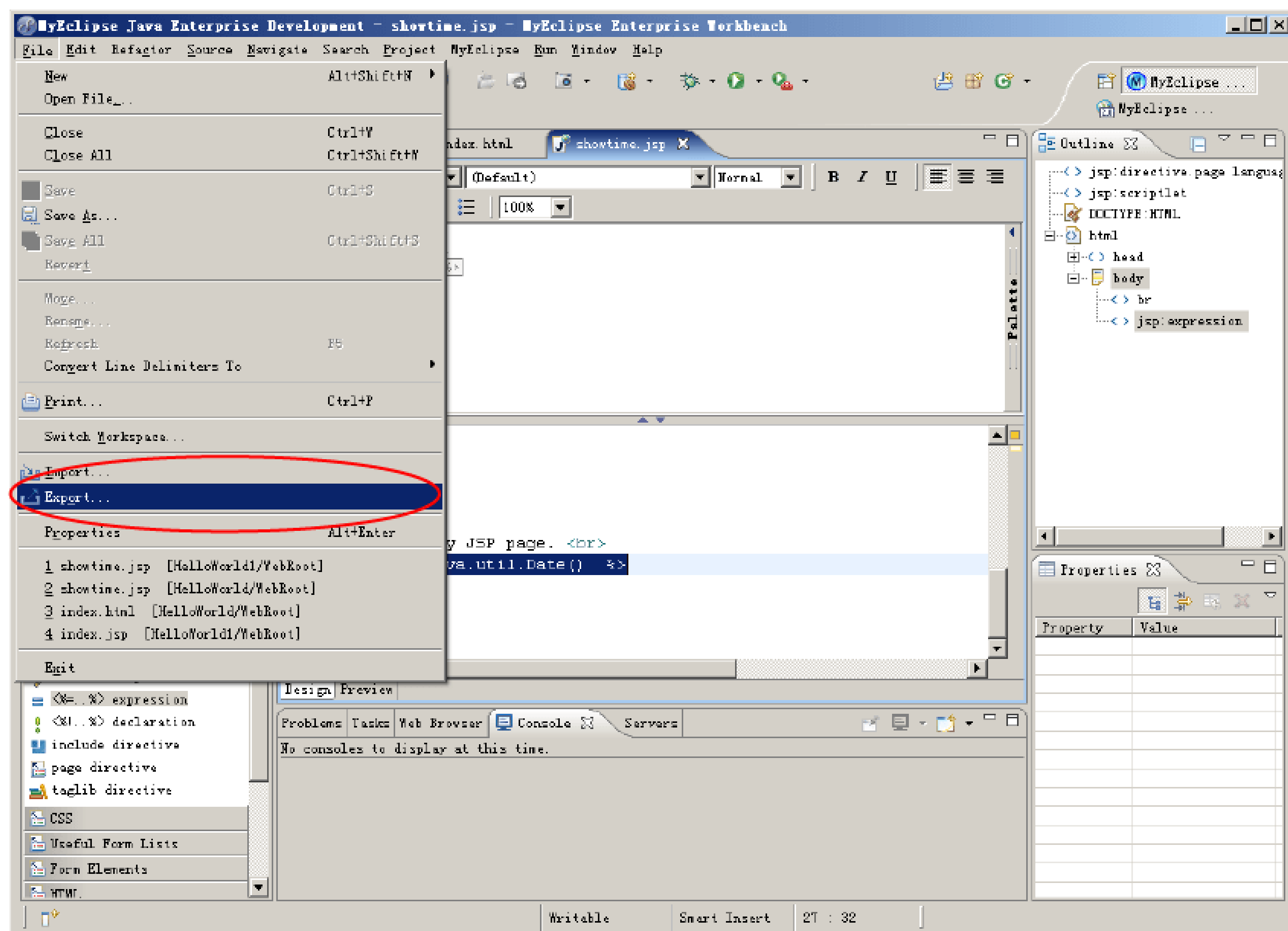
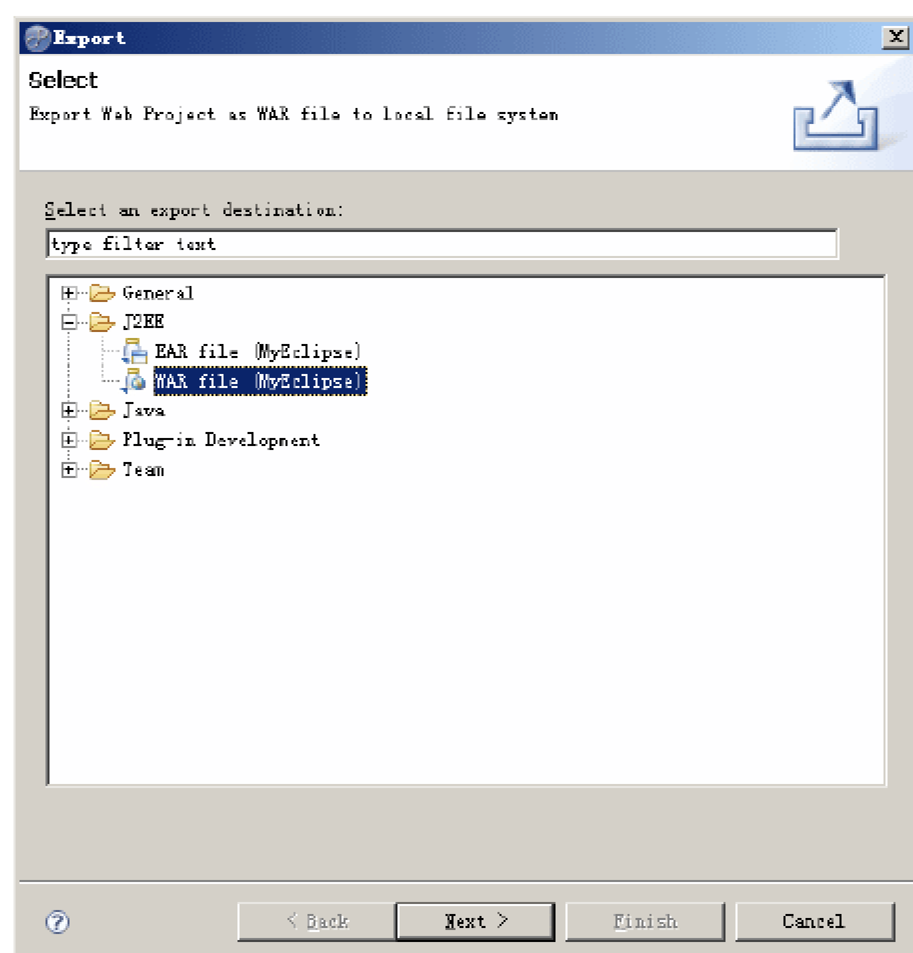


图 8-40 发布 Java Web 工程

❷ 在打开的如图 8-41 所示的选择 Export 类型的窗口中，可以选择供用户所要导出的类型。当然选择 WAR file 类型，然后单击 Next 按钮。

❸ 单击 Next 按钮后，弹出如图 8-42 所示的选择 Export 工程名的窗口，选择 HelloWorld1 工程，如图 8-42 的标记部分；并选择 HelloWorld1.war 文件的发布路径，然后单击 Finish 按钮完成工程的发布。

📖**点评与拓展：**以上两节提供了开发 Web 工程的基本步骤，无论多么复杂的 Web 工程，其开发步骤都类似于这个 HelloWorld1 工程。



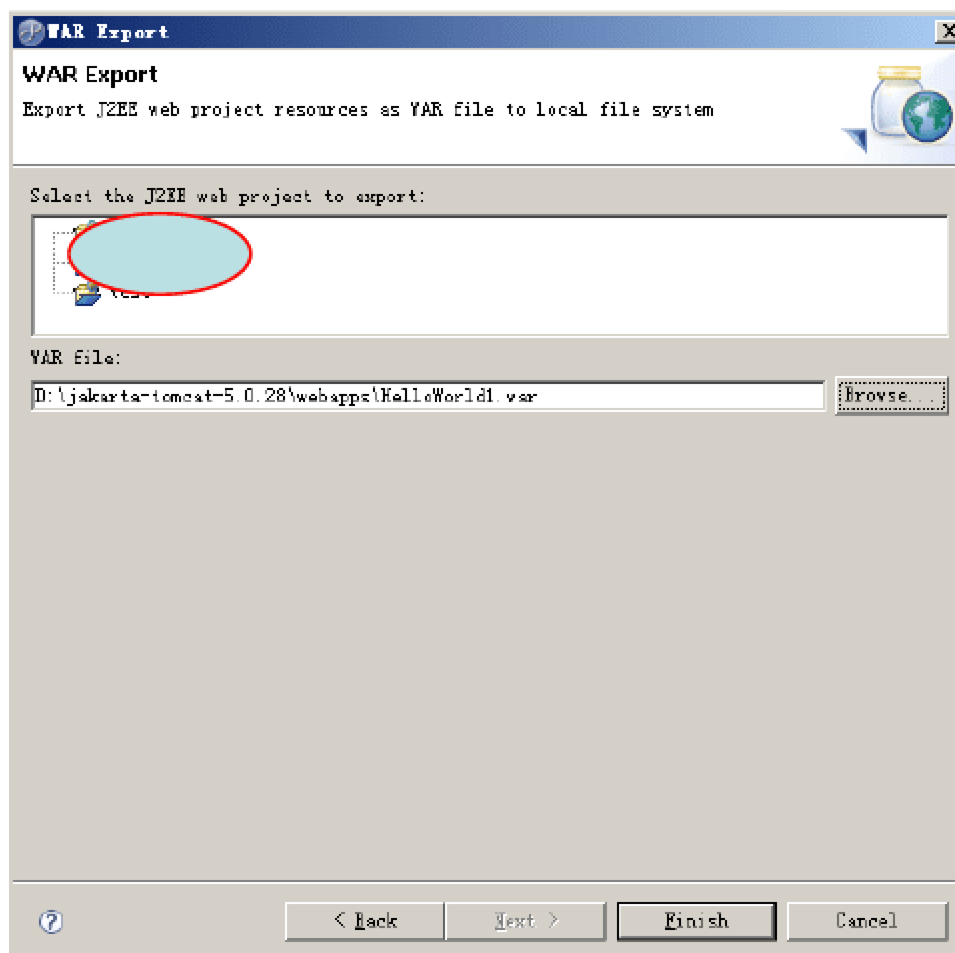


图 8-41　选择 Export 类型　　　　图 8-42　选择 Export 的工程名完成发布

8.4.3　部署 HelloWorld1.war 文件

将 Web 工程发布成.war 文件后，接下来介绍如何将上节生成的 HelloWorld1.war 文件部署到 Tomcat 服务器中，并且访问这个 Web 工程。

❶ 利用 SSH Secure Shell Client 软件提供的 FTP 传输工具将 HelloWorld1.war 上传到服务器的 $CATALINA_HOME/webapps 目录下，如图 8-43 所示。

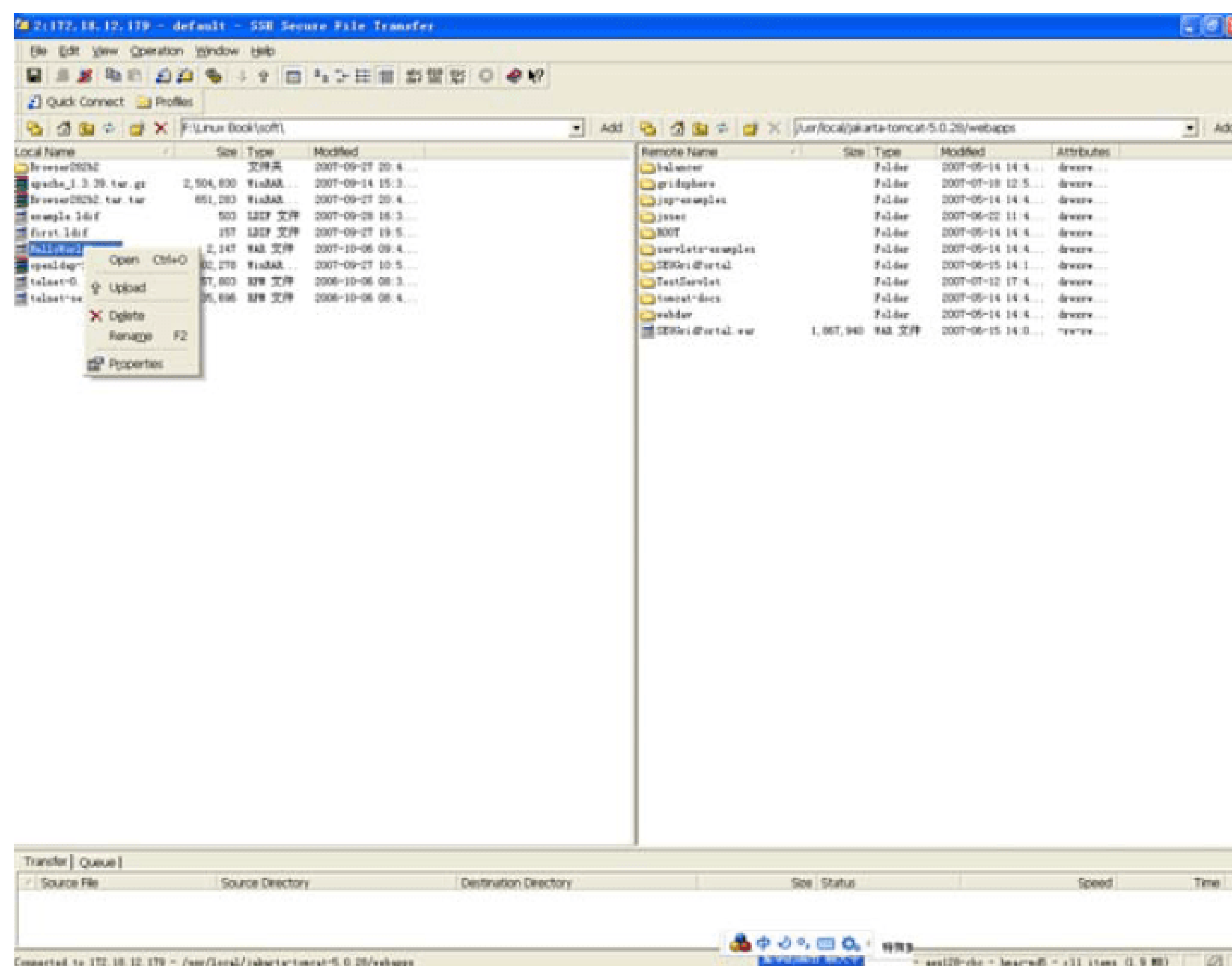


图 8-43　上传 HelloWorld1.war

❷ 如图 8-44 所示，在$CATALINA_HOME/webapps 目录下出现了 HelloWorld1.war 这个文件。启动 Tomcat 服务器，过了十几秒钟后，Tomcat 会把 HelloWorld1.war 自动解压，生成 HelloWorld1 文件夹，这个文件夹就是 HelloWorld1 这个 Web 工程的根目录，有着与开发时一样的目录结构，首页为 index.jsp，还包含了新添加的 showtime.jsp 页面。

```
[root@seugrid3 webapps]# ls
balancer  gridsphere  HelloWorld1.war  jsp-examples  jssec  ROOT  servlets-examples  SEUGridPortal
[root@seugrid3 webapps]# cd ..
[root@seugrid3 jakarta-tomcat-5.0.28]# ./bin/startup.sh
Using CATALINA_BASE:   /usr/local/jakarta-tomcat-5.0.28
Using CATALINA_HOME:   /usr/local/jakarta-tomcat-5.0.28
Using CATALINA_TMPDIR: /usr/local/jakarta-tomcat-5.0.28/temp
Using JAVA_HOME:       /usr/jdk1.5.0_04
[root@seugrid3 jakarta-tomcat-5.0.28]# ls webapps/
balancer  gridsphere  HelloWorld1  HelloWorld1.war  jsp-examples  jssec  ROOT  servlets-examples
[root@seugrid3 jakarta-tomcat-5.0.28]#
```

图 8-44　Tomcat 自动解压 HelloWorld1.war

❸ 在浏览器地址栏输入http://IP:port/HelloWorld1格式的地址，就能访问 index.jsp 这个首页，本例输入的是http://172.18.12.179:9080/HelloWorld1，结果如图 8-45 所示。单击网页上 show current time 栏目，就自动跳转到 showtime.jsp 这个页面，这个页面的功能是显示当前的系统时间，结果如图 8-46 所示。

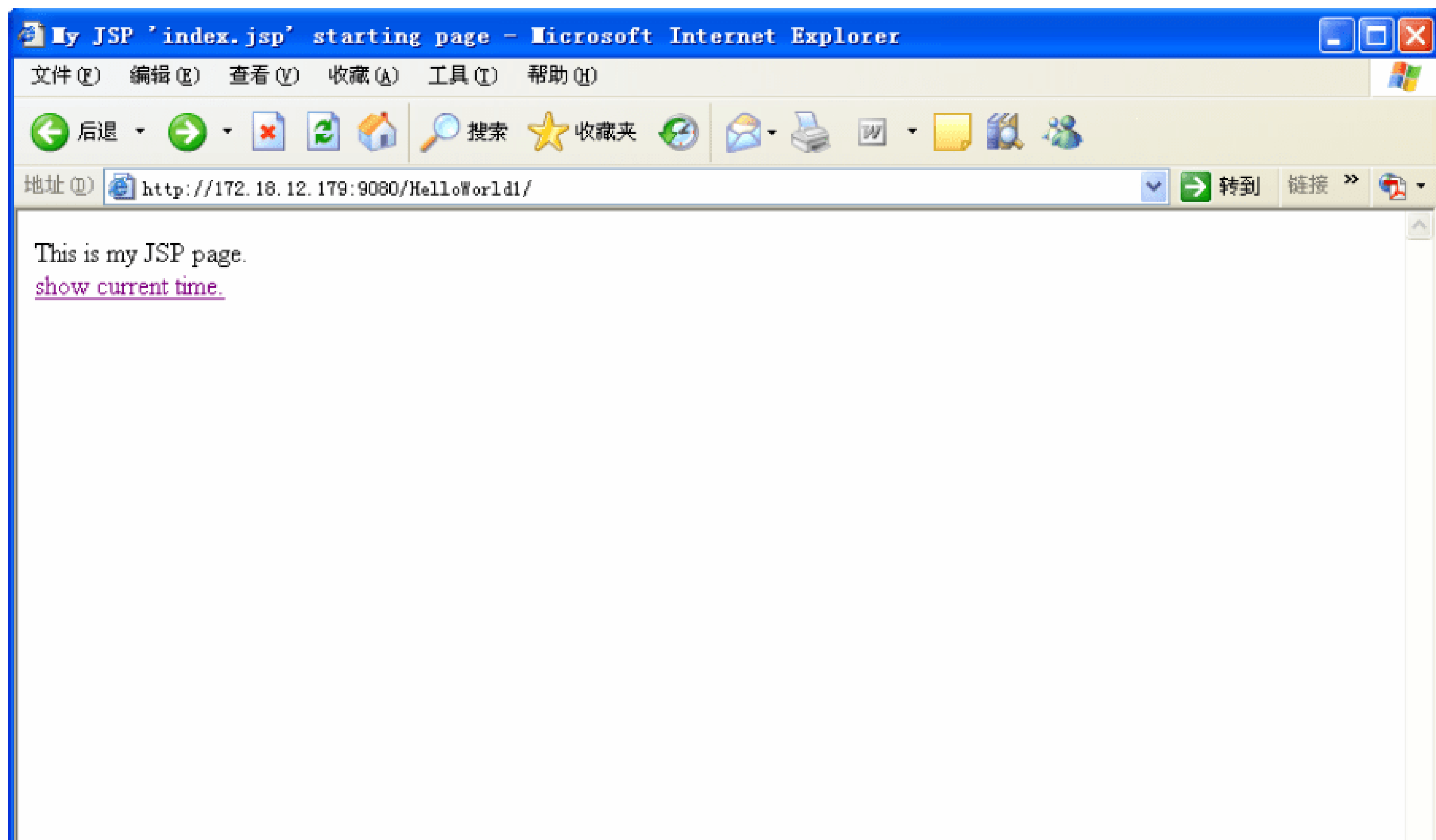


图 8-45　访问 index.jsp 页面

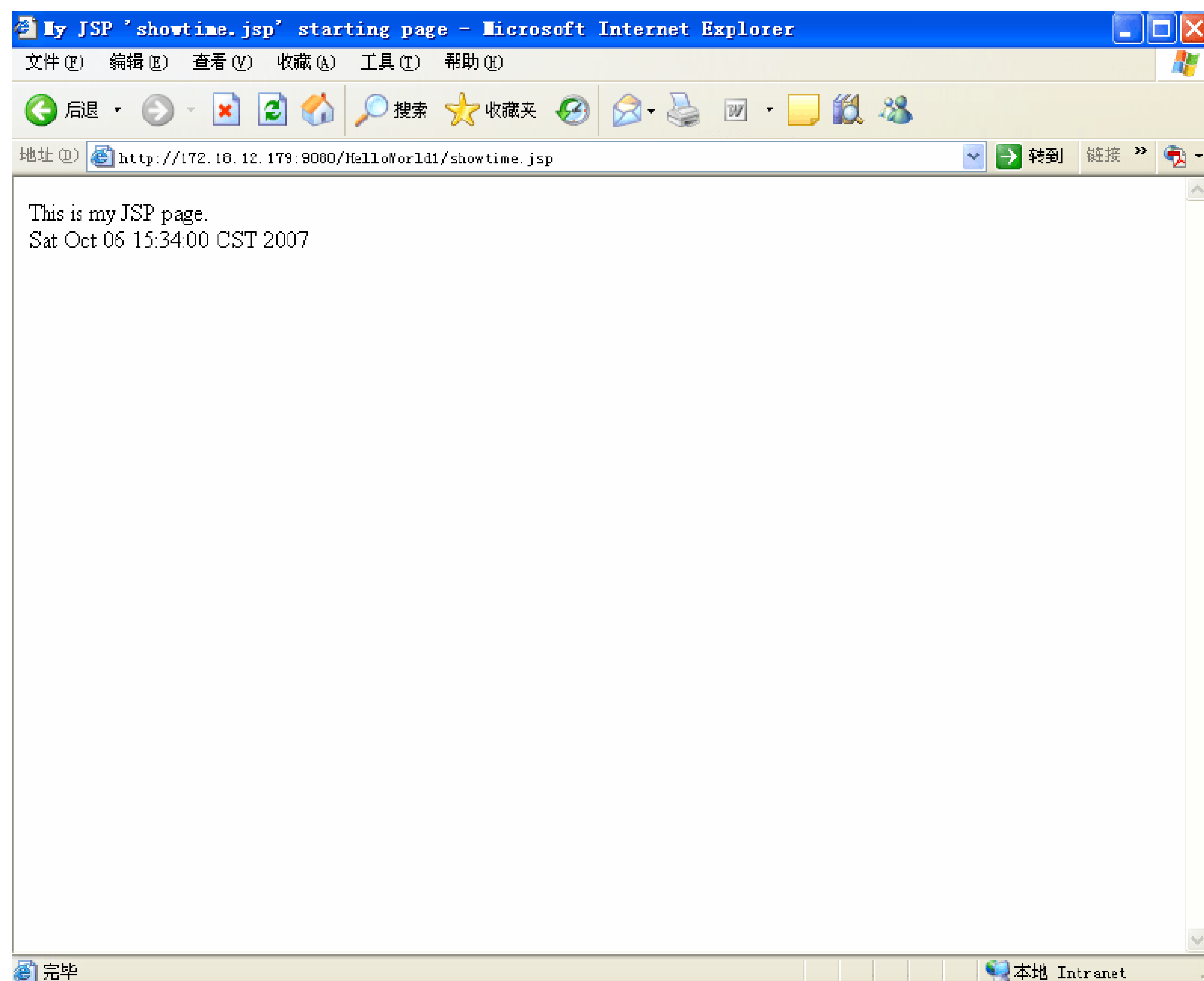


图 8-46　访问 showtime.jsp 页面

点评与拓展：本节讲述的.war 包在 Tomcat 服务器中的部署步骤，适用于任何 Web 工程的部署，具有普遍性。

8.5　本章小结

架设 Web 服务器是建构 Internet 和 Intranet 非常关键的工作，本章分别介绍了 Apache 和 Tomcat 服务器的安装、配置和启动。与 Apache 相比，Tomcat 扩展了对 JSP 动态网页的支持，因此，本章通过一个简易的 Web 工程讲述了利用 MyEclipse 创建 Web 工程的基本步骤，并将其发布成.war 文件，然后讲述了如何将这个.war 文件部署到 Tomcat 服务器中，该方法具有普遍适用性，可以扩展到更为复杂的 Web 工程应用中。

第 9 章　FTP 服务器的配置与架设

FTP(File Transfer Protocal，文件传输协议)服务是最基本的网络服务之一，用于控制 Internet 上文件的双向传输。用户通过它可以把自己的 PC 与世界各地所有运行FTP 协议的服务器相连，实现与远程服务器的数据互传。本章主要介绍 FTP 的基本工作原理，以及使用 vsftpd 服务器软件架设 FTP 服务器的方法，最后简单介绍 Linux 系统下 FTP 的客户端程序 gftp。

通过本章的学习，读者应掌握以下内容：

- ✧ FTP 的工作原理
- ✧ FTP 的两种连接模式
- ✧ vsftpd 的安装和配置
- ✧ gftp 的安装和配置

9.1　FTP 服务概述

9.1.1　FTP 服务简介

FTP 的主要功能就是让用户连接上一个远程计算机(这些计算机上运行着 FTP 服务器程序)，查看远程计算机有哪些文件，然后要么把文件从远程计算机上复制到用户计算机，要么把用户计算机的文件传到远程计算机去。目前，FTP 虽有渐被万维网服务取代之势，但由于其管理的简单性和双向传输特性，使其仍得到广泛应用。

下面详细介绍一下 FTP 协议(文件传输协议)。 通常用户连接上 Internet 的首要目的是实现资源共享，而文件传输是实现资源共享的非常重要的一个手段。早期在 Internet 上实现文件传输，并不是一件容易的事，因为 Internet 是一个非常复杂的计算机环境，有 PC，有工作站，有 MAC，有大型机，连接于其上的计算机成千上万，而这些计算机可能运行不同的操作系统，有运行 UNIX 的服务器，也有运行 DOS、Windows 的 PC 和运行 MacOS 的苹果机，等等。对于各种操作系统之间的文件交流问题，需要建立一个统一的文件传输协议，这就是所谓的 FTP。支持 FTP 协议的服务器就是所谓 FTP 服务器。基于不同的操作系统就有不同的 FTP 应用程序，而所有这些应用程序都遵守同一种协议，这样用户就可以把自己的文件传送给别人，或者从其他的用户环境中获得文件。

9.1.2　FTP 工作原理

FTP 采用“客户/服务器”方式，用户端要在自己的本地主机上安装 FTP 客户程序。FTP 客户程序有字符界面和图形界面两种。字符界面的 FTP 命令复杂繁多，图形界面的 FTP 客户程序操作简洁方便。用户通过一个支持 FTP 协议的客户机程序，连接到在远程主机上的 FTP 服务器程序。用户通过客户机程序向服务器程序发出命令，服务器程序执行用户所发出的命令，并将执行的结果返回到客户机。

客户端和服务器使用 TCP 建立连接，建立连接时，客户端和服务器都要各自打开一个 TCP 端口。FTP 服务器有两个预分配的端口号(21 和 20)，端口 21 用于发送和接收 FTP 控制信息。FTP 服务器通过侦听 21 端口判断是否有客户连接请求。一个会话连接建立后，21 端口在连接会话期间将始终打开。20 端口用于发送和接收数据，该端口只在数据传输时打开，数据传输结束后即关闭。FTP 客户端启动 FTP 客户机程序后，动态分配其端口号(范围为 1024～65535)。一个 FTP 会话开始后，客户机程序打开一个控制端口(比如取 1027)，该端口连接服务器的 21 端口。要传输数据时，客户端打开连接到服务器 20 端口的另一个端口(比如取 1028)。每次开始传输文件时，客户机程序都会打开一个新的数据端口，传输结束后再自动释放掉。如图 9-1 所示，FTP 的具体工作步骤如下。

❶ FTP 客户端发送请求，系统动态分配一个控制端口 1027。

❷ FTP 服务器端口 21 侦听到请求后，与客户端端口 1027 建立会话连接。

❸ 若要传输数据，FTP 客户端动态打开一个连接到 FTP 服务器 20 端口的新端口 1028，这样两个端口间就可以进行数据传输。

❹ 数据传输完毕后，数据端口 1028 和服务器 20 端口自动关闭，会话连接继续保持。

❺ FTP 客户端与服务器端断开连接后，客户端动态分配的端口 1027 自动释放。

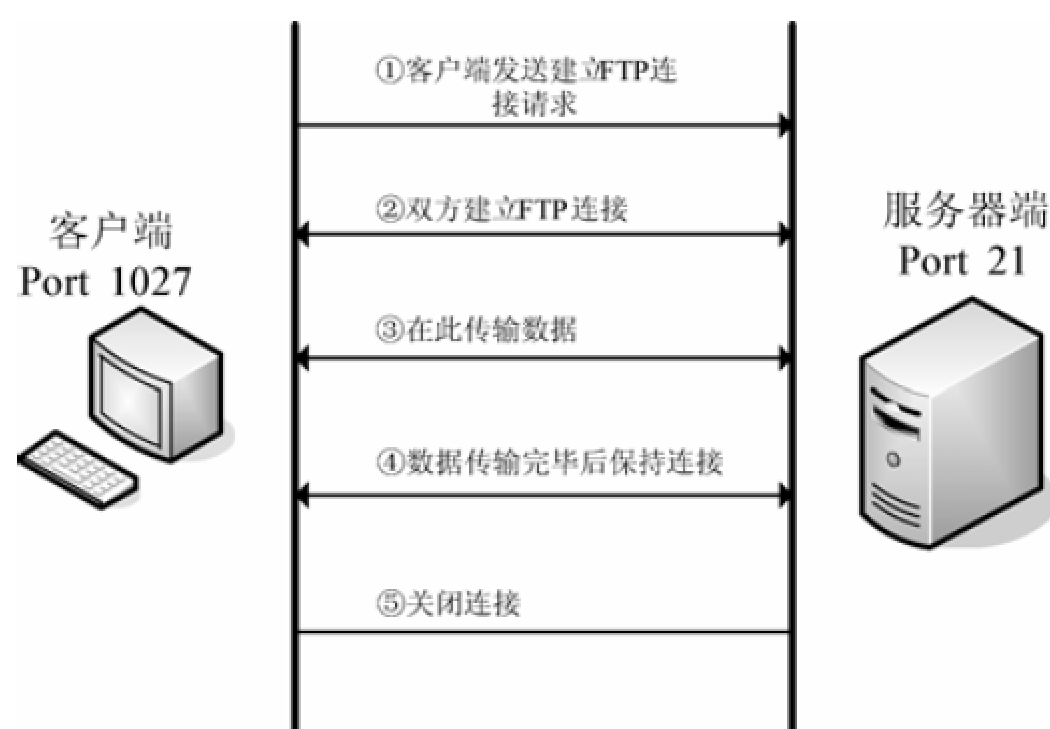


图 9-1　FTP 服务工作原理

9.1.3　FTP 的两种连接模式

FTP 的连接模式有两种：PORT(主动模式)和 PASV(被动模式)。主动模式的连接过程是：

客户端动态地选择一个端口(这个端口号一定是 1024 以上的，因为 1024 以前的端口都已经预先被定义好了)向服务器端的 FTP 端口(默认是 21)发送连接请求，服务器接受请求，建立一个控制连接。当需要传送数据或列出服务器的文件列表时(通常使用 ls 或 dir 命令)，客户端通过控制连接告诉服务器(使用 PORT 命令)：“我已经打开了 XX 端口，请你过来连接。”于是服务器使用 20 端口向客户端的 XX 端口发送连接请求，建立一条数据连接来传送数据。

被动模式是指，客户端首先使用与主动连接模式相同的方法与服务器建立控制连接。当需要传送数据时，客户端通过控制连接告诉服务器(使用 PASV 命令)“我要连接你的 XX 端口，请问是否空闲”，如果恰好该端口空闲，服务器会告诉客户端：“你请求的端口空闲，可以建立连接(ACK 确认信息)。”否则服务器会说“该端口已经占用，请换个端口(UNACK 信息)”。如果客户端得到的是空闲的提示，就可以利用该端口建立连接，否则就得换个端口重新尝试，这也就是所谓的连接建立的协商过程。

区分主动模式和被动模式的主要目的是：并不是所有的 FTP 服务都支持这两种连接模式，例如：微软自带的 FTP 命令客户端就不支持被动模式，如果设置错了客户将无法连接。

9.2 使用 vsftpd 架设 FTP 服务器

vsftpd (very secure FTP daemon)是针对传统 FTP 软件安全性不足而设计出来的，其目的就是建构一个具有良好安全性保证的 FTP 服务器。

那么 vsftpd 是如何来保证 FTP 服务器的安全的呢？首先， vsftpd 考虑了 Linux 系统进程的权限(privilege)所带来的安全问题。一个进程由惟一的进程号标识，叫做 PID，PID 拥有的权限等级越高，它能够进行的任务就越多。举例来说，使用 root 身份所触发的 PID 通常拥有可以进行任何工作的权限等级。因此，被网络黑客所控制的进程的 PID 越高，所产生的危险性就越大，因为这可能导致黑客能够修改系统的任意其他进程或文件。vsftpd 正是基于这种隐患设计的，它将自己 PID 的权限尽量设低以提高系统的安全性。

其次，vsftpd 还支持 chroot 功能，chroot 可以将 FTP 登录用户限制在特定目录下，并将这个特定目录变成该用户的根目录，这样与该目录没有关系的其他目录就不会被误用了。

下面分别在两种场景下讲述如何使用 vsftpd 来架设 FTP 服务器。

9.2.1 架设内部 FTP 服务器

应用实例导航——A 公司局域网 FTP 服务器架设

※场景呈现

A 公司需要在内部局域网架设 FTP 服务器，以方便内部员工的资源共享。由于内部局域网是信任度相对较高的环境，因此，该 FTP 服务器允许匿名登录及系统用户登录。对于匿名登录的用户将其家目录设置为/var/ftp，且无上传权限；对于系统用户登录，其家目录

设置为/home 下该用户的根目录。

另外，为了方便 FTP 服务器的管理，FTP 服务器要能够通过配置/etc/vsftpd.user_list 文件来实现不允许登录的用户。

※技术要领

(1) vsftpd 的启动。
(2) 配置允许匿名用户登录。
(3) 配置允许系统用户登录。
(4) 使/etc/vsftpd.user_list 设置生效。

首先通过实例场景中这样一个信任度较高、用户权限可以适当放宽的环境来讲述 vsftpd 的启动和一些基本的配置，这样可以使读者的注意力更多地放到 vsftpd 配置文件的结构和关系、基本配置的意义上，而不必过多地纠缠于用户的配置权限。

❶ Fedora 6.0 默认安装了 vsftpd，如图 9-2 所示，可以使用下面命令查看：

```
rpm -qa | grep ftp
```

```
[root@seugrid3 local]# rpm -qa | grep ftp
ftp-0.17-32.1.2
vsftpd-2.0.4-1.2
lftp-3.4.2-5
[root@seugrid3 local]#
```

9-2　查看 vsftpd 是否安装

❷ vsftpd 最主要的配置文件为 vsftpd.conf，存放在/etc/vsftpd 目录下。严格来说，vsftpd.conf 设定了所有与 vsftpd 有关的参数，其他设定文件都附属于此 vsftpd.conf 文件。这个文件的设定是以与 bash 变量设定相同的方式来处理的， 也就是“参数=设定值”来设定的，注意，等号两边不能有空格符。下面命令将提供详细的 vsftpd.conf 参数说明：

```
man 5 vsftpd.conf
```

❸ 接着讲述实现本例的 vsftpd.conf 主要配置，用 vi 命令打开它，如图 9-3 所示。图中的前两条划线语句是默认值，无需修改，只需确定。后面的三条语句是需要读者自行添加进去的。下面分别对这几条语句进行解释：

```
anonymous_enable=YES
```

该值默认是 YES，说明允许匿名用户登录 FTP 服务器；若将其改为 NO，则匿名用户无法登录。

```
local_enable=YES
```

当该值设定为 YES 时，允许/etc/passwd 内的系统账户登录 FTP 服务器。

```
userlist_enable=YES
```

该值说明是否利用 vsftpd 来处理某些不允许登录的用户。

```
userlist_deny=YES
```

当 userlist_enable 设为 YES 时，此项和下一项才有意义。该项设为 YES 时，如果使用者账户被列入到某个文件，他将无法登入 FTP 服务器。

```
userlist_file=/etc/vsftpd.user_list
```

该项设定了 vsftpd 阻止或接受用户所在文件的路径，与上一项相关联，而且指定的文件必须已经被创建。

```
[root@seugrid3 local]# vi /etc/vsftpd/vsftpd.conf
# Example config file /etc/vsftpd/vsftpd.conf
#
# The default compiled in settings are fairly paranoid. This sample file
# loosens things up a bit, to make the ftp daemon more usable.
# Please see vsftpd.conf.5 for all compiled in defaults.
#
# READ THIS: This example file is NOT an exhaustive list of vsftpd options.
# Please read the vsftpd.conf.5 manual page to get a full idea of vsftpd's
# capabilities.
#
# Allow anonymous FTP? (Beware - allowed by default if you comment this out).
anonymous_enable=YES
#
# Uncomment this to allow local users to log in.
local_enable=YES
#
# Uncomment this to enable any form of FTP write command.
write_enable=YES
#
# Default umask for local users is 077. You may wish to change this to 022,
# if your users expect that (022 is used by most other ftpd's)
local_umask=022

userlist_enable=YES
userlist_deny=YES
userlist_file=/etc/vsftpd.user_list
```

图 9-3　允许匿名和系统用户的配置

❹ 启动 vsftpd 的方法与一般服务类似，它默认在 21 号端口侦听，两条命令分列于下，结果如图 9-4 所示。

```
/etc/init.d/vsftpd start
netstat -tulnp | grep 21
```

```
[root@seugrid3 ~]# /etc/init.d/vsftpd start
为 vsftpd 启动 vsftpd:                                    [确定]
[root@seugrid3 ~]# netstat -tulnp | grep 21
tcp        0      0 0.0.0.0:21            0.0.0.0:*              LISTEN      5428/vsftpd
udp        0      0 0.0.0.0:32770         0.0.0.0:*                          2158/avahi-daemon:
udp        0      0 0.0.0.0:5353          0.0.0.0:*                          2158/avahi-daemon:
udp        0      0 :::32771              :::*                               2158/avahi-daemon:
[root@seugrid3 ~]#
```

图 9-4　启动 vsftpd

❺ 假设存在一个系统用户 ftptest，可以测试它能否登录刚刚启动的 FTP 服务器，只要输入下面命令连接本机 FTP 服务器：提示输入用户名，输入 ftptest，提示输入密码，输入 ftptest 用户的密码，如图 9-5 所示。

```
ftp localhost
```

我们可以看到，ftptest 用户成功登录，并且家目录就为 ftptest 的用户根目录，如图 9-5 划线部分所示。

```
[root@seugrid3 local]# ftp localhost
Connected to localhost.
220 (vsFTPd 2.0.4)
530 Please login with USER and PASS.
530 Please login with USER and PASS.
KERBEROS_V4 rejected as an authentication type
Name (localhost:root): ftptest
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> pwd
257 "/home/ftptest"
ftp>
```

图 9-5　测试 FTP 服务器

6 接下来，测试/etc/vsftpd.user_list 文件是否能够生效，使用 vi 命令编辑 vsftpd.user_list 文件，添加一行 ftptest，如图 9-6 所示。

```
[root@seugrid3 local]# vi /etc/vsftpd.user_list
tian
ftptest
```

图 9-6　编辑 vsftpd.user_list 文件

然后，按步骤(5)登陆 FTP 服务器，系统会提示 Permission denied，即登录失败，说明上面 vsftpd.user_list 文件已经生效，结果如图 9-7 所示。

```
[root@seugrid3 local]# ftp localhost
Connected to localhost.
220 (vsFTPd 2.0.4)
530 Please login with USER and PASS.
530 Please login with USER and PASS.
KERBEROS_V4 rejected as an authentication type
Name (localhost:root): ftptest
530 Permission denied.
Login failed.
ftp>
```

图 9-7　vsftpd.user_list 生效

点评与拓展：由于内部 FTP 服务器处于信任度较高的环境，所以配置工作相对简单。读者只要掌握好 vsftpd.conf 的配置命令即可熟练操作。

9.2.2 架设实用 FTP 服务器

应用实例导航——A 公司实用 FTP 服务器架设

※场景呈现

随着 A 公司业务流程的复杂化，与外部的资源共享和交流越来越频繁，A 公司准备将内部的 FTP 服务器改造成实用的 FTP 服务器，使其不仅适合于内部局域网这种信任度相对较高的环境，而且对外部 Internet 网络开放。具体要求描述如下：

(1) 用户登录 FTP 服务器时显示一段欢迎信息；
(2) FTP 使用本地时间代替格林威治时间；
(3) 将用户所创建的目录、文件的权限设置为 775；
(4) 建立严格的可使用 FTP 服务器的用户列表；
(5) 将登录 FTP 的某些系统用户的访问范围限制在他的根目录下；
(6) 将下载的最大带宽限制在 100KB/s；
(7) 不允许同一个 IP 地址建立两个以上的连接。

※技术要领

(1) 欢迎信息的设定。
(2) 权限掩码(umask)的配置。
(3) chroot 的设置。

本实例应用场景是一个信任度相对较低、对外部网络开放的环境，配置要求也相对较多。具体的实现过程如下。

1. 建立 FTP 欢迎信息

通常情况下，用户登录到 FTP 服务器时，会看到一条类似于公告的信息，通常的内容是欢迎你访问该 FTP，并且会公布下载不同类型文件的账号和密码。本节介绍这种欢迎信息的设定方法。

❶ 首先新建 welcome.txt 文件，这个纯文本文件记录了该 FTP 服务器所需要显示的欢迎信息，如图 9-8 所示。

❷ 打开/etc/vsftpd/vsftpd.conf 文件，添加如图 9-9 标注部分的语句。其中 banner_file 指定了 FTP 服务器读取的欢迎信息文件的路径，比如本例将上一步新建的 welcome.txt 文件放置到/etc/vsftpd 目录中。

```
[root@seugrid3 ~]# vi /etc/vsftpd/welcome.txt
*************************************
*                                   *
*          欢迎光临vsftpd FTP         *
*                                   *
*  该帐号仅用于下载软件、学习类资料      *
*                                   *
*          不能下载娱乐内容            *
*                                   *
*   所有娱乐相关资源请用匿名登陆        *
*                                   *
*************************************
```

图 9-8　新建 welcome.txt 文件

```
userlist_enable=YES
userlist_deny=YES
userlist_file=/etc/vsftpd.user_list
banner_file=/etc/vsftpd/welcome.txt
```

图 9-9　vsftpd.conf 文件中的 banner_file 设置

❸ 设置完后，如果再次登录 FTP 服务器时，就会显示 welcome.txt 文件的欢迎信息，如图 9-10 所示。

```
[root@seugrid3 ~]# ftp localhost
Connected to localhost.
220-*************************************
220-*                                   *
220-*          欢迎光临vsftpd FTP         *
220-*                                   *
220-*  该帐号仅用于下载软件、学习类资料      *
220-*                                   *
220-*          不能下载娱乐内容            *
220-*                                   *
220-*   所有娱乐相关资源请用匿名登陆        *
220-*                                   *
220-*************************************
220
530 Please login with USER and PASS.
530 Please login with USER and PASS.
KERBEROS_V4 rejected as an authentication type
Name (localhost:root): ftptest
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd /usr/local
250 Directory successfully changed.
ftp> pwd
257 "/usr/local"
```

图 9-10　测试欢迎信息

2．设置创建目录和文件的默认权限

用户在自己的根目录下具有创建目录和文件的默认权限，那么如何设置所创建的目录和文件的默认权限呢？vsftpd 是通过权限掩码(umask)的方式来实现的，本节介绍何谓权限掩码，以及如何设置 vsftpd 的权限掩码。

从 1.3.2 节“文件和目录操作”的介绍知道 Linux 文件的权限是由 3 位八进制数字所组成，初始的默认权限是 777。权限掩码表示从现有权限中需要减去的权限，如果 umask 值设定为 077，那么所创建的目录和文件的默认权限就为 777-077=700。

实例要求将所创建的目录、文件的权限设置为 775，那么 umask=777-775=002，在 /etc/vsftpd/vsftpd.conf 文件中添加 local_umask=002 即可，如图 9-11 所示。

```
# Uncomment this to allow local users to log in.
local_enable=YES
#
# Uncomment this to enable any form of FTP write command.
write_enable=YES
#
# Default umask for local users is 077. You may wish to change this to 022,
# if your users expect that (022 is used by most other ftpd's)
local_umask=002

userlist_enable=YES
userlist_deny=YES
userlist_file=/etc/vsftpd.user_list
banner_file=/etc/vsftpd/welcome.txt
```

图 9-11　vsftpd.conf 文件中的 local_umask 设置

3．chroot 相关设置

上面曾经提及 vsftpd 之所以安全的一个主要原因就是可以将登录的系统用户限制在他自己的根目录下，这个设置就是通过 chroot 来实现的。下面介绍 chroot 的设置方法。

❶ 打开/etc/vsftpd/vsftpd.conf 文件，添加如图 9-12 标注部分的语句，两条语句解释如下:

chroot_list_enable=YES

启动将某些系统用户限制在其根目录下，默认为 NO。

chroot_list_file=/etc/vsftpd.chroot_list

当 chroot_local_user 设置为 YES 方才生效，这时 vsftpd.chroot_list 文件中列出的用户将无法离开其根目录。

```
# Default umask for local users is 077. You may wish to change this to 022,
# if your users expect that (022 is used by most other ftpd's)
local_umask=002

userlist_enable=YES
userlist_deny=YES
userlist_file=/etc/vsftpd.user_list
banner_file=/etc/vsftpd/welcome.txt

chroot_list_enable=YES
chroot_list_file=/etc/vsftpd.chroot_list
```

图 9-12　vsftpd.conf 文件中的 chroot 设置

❷ 打开/etc/vsftpd.chroot_list 文件，假设现在添加 ftptest 用户，如图 9-13 所示。在没有设置 chroot 功能前，当 ftptest 用户登录 FTP 服务器时，输入 pwd 将显示/home/ftptest 目录，并且能够进入/usr/local 目录，如图 9-13 所示。在 vsftpd.chroot_list 文件中添加 ftptest 后，重启 vsftpd 服务，再用 ftptest 用户登录 FTP 服务器时，输入 pwd 将显示 "/"，意思即为根目录；而且无法进入/usr/local 目录，如图 9-14 标注部分所示。

```
[root@seugrid3 ~]# vi /etc/vsftpd.chroot_list
ftptest
~
```

图 9-13　编辑 vsftpd.chroot_list 文件

```
[root@seugrid3 ~]# ftp localhost
Connected to localhost.
220-*****************************************
220-*                                       *
220-*           欢迎光临vsftpd FTP           *
220-*                                       *
220-*   该帐号仅用于下载软件、学习类资料     *
220-*                                       *
220-*           不能下载娱乐内容             *
220-*                                       *
220-*    所有娱乐相关资源请用匿名登陆        *
220-*                                       *
220-*****************************************
220
530 Please login with USER and PASS.
530 Please login with USER and PASS.
KERBEROS_V4 rejected as an authentication type
Name (localhost:root): ftptest
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> pwd
257 "/"
ftp> cd /usr/local
550 Failed to change directory.
ftp> bye
221 Goodbye.
[root@seugrid3 ~]#
```

图 9-14　测试 chroot 功能

4．其他的 vsftpd 设置

本节讲述实例中剩下的一些设置工作，比如，将 FTP 时间设定为本地时间、将下载的最大带宽限制在 100KB/s、不允许同一个 IP 地址建立两个以上的连接以及建立严格的可使用 FTP 服务器的用户列表等。

❶ vsftpd 服务默认的时间是格林威治时间，与北京时间有 6 个小时左右时差，如果将 use_localtime 改成 YES，则 FTP 使用本地时间，如图 9-15 所示。

```
local_umask=002

userlist_enable=YES
userlist_deny=YES
userlist_file=/etc/vsftpd.user_list


chroot_list_enable=YES
chroot_list_file=/etc/vsftpd.chroot_list


use_localtime=YES
banner_file=/etc/vsftpd/welcome.txt
```

图 9-15　vsftpd.conf 文件中的 use_localtime 设置

❷ 通常情况下，FTP 服务器都需要限制用户下载的最大带宽，否则有限的带宽资源将很容易被几个用户耗尽，影响其他用户的使用，也会加重 FTP 服务器的负载。同时，一般的 FTP 服务器不能容忍一个 IP 地址同时建立多个连接，因为这样会变相地增大下载带宽。图 9-16 演示了 vsftpd 设置性能方面的几个参数，解释如下:

```
local_max_rate=100000
```

限制用户下载的最大带宽，单位是 B/s。

```
max_clients=10
```

最多 10 个 IP 同时连接 FTP 服务器。

```
max_per_ip=1
```

每个 IP 地址只允许建立一个连接。

```
local_umask=002

userlist_enable=YES
userlist_deny=YES
userlist_file=/etc/vsftpd.user_list


chroot_list_enable=YES
chroot_list_file=/etc/vsftpd.chroot_list


use_localtime=YES
banner_file=/etc/vsftpd/welcome.txt

local_max_rate=100000

max_clients=10
max_per_ip=1
```

图 9-16　FTP 限速和限制 IP 连接数的设置

❸ 上面介绍了如何设定不允许登录的用户，不过有时还需要设定有限的几个能够登录的用户，设置之外的都不允许登录，这时也是用 userlist 的几个参数，区别在于，将 userlist_deny 改成 NO，如图 9-17 标注部分所示。这样，只有在文件/etc/vsftpd.user_list 中列出的用户才能登录 FTP 服务器。

```
local_umask=002

userlist_enable=YES
userlist_deny=NO
userlist_file=/etc/vsftpd.user_list


chroot_list_enable=YES
chroot_list_file=/etc/vsftpd.chroot_list
```

图 9-17　建立可使用 FTP 的用户列表

点评与拓展： 在 vsftpd.user_list 和 vsftpd.chroot_list 等文件设置用户账户时，如果有多个账户，需要分行写入，每行写一个用户的账户，这样 vsftpd 才能识别这些不同的用户。

9.3　gftp FTP 客户端程序简介

Linux 系统提供了 shell 命令登录 FTP，但是这种方式不直观，用户希望有像 Windows 系统下 CuteFTP 或 LeapFTP 一样的工具来实现 FTP 文件传输。gftp 就是这样一款软件，它提供了图形化界面，使用起来十分直观方便，本节介绍 gftp 的安装和使用。

❶ 通过下载或本书光盘得到安装包：gftp-2.0.18.tar.tar，利用下面命令解压缩，得到/usr/local/gftp-2.0.18 安装目录。

```
tar -zxvf /usr/local/gftp-2.0.18.tar.tar
```

❷ 进入解压后得到的目录为/usr/local/gftp-2.0.18，输入下面命令进行安装配置，假设最后的安装目录为/usr/local/gftp，如图 9-18 所示。

```
./configure -prefix=/usr/local/gftp
```

```
[root@seugrid3 local]# cd gftp-2.0.18/
[root@seugrid3 gftp-2.0.18]# ./configure --prefix=/usr/local/gftp
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking build system type... i686-pc-linux-gnu
```

图 9-18　安装配置 gftp 安装包

❸ 接下来就是编译 gftp 安装包，输入下面命令进行编译，如图 9-19 所示。

```
make
```

```
[root@seugrid3 gftp-2.0.18]# make
make  all-recursive
make[1]: Entering directory `/usr/local/gftp-2.0.18'
Making all in intl
make[2]: Entering directory `/usr/local/gftp-2.0.18/intl'
make[2]: Nothing to be done for `all'.
make[2]: Leaving directory `/usr/local/gftp-2.0.18/intl'
Making all in docs
```

图 9-19　编译 gftp 安装包

❹ 最后是安装 gftp 安装包，输入下面命令进行安装，如图 9-20 所示。

```
make install
```

```
[root@seugrid3 gftp-2.0.18]# make install
Making install in intl
make[1]: Entering directory `/usr/local/gftp-2.0.18/intl'
if { test "gftp" = "gettext-runtime" || test "gftp" = "gettext-tools"; } \
   && test 'no' = yes; then \
  /bin/sh ../../mkinstalldirs /usr/local/gftp/lib /usr/local/gftp/include; \
  /usr/bin/install -c -m 644 libintl.h /usr/local/gftp/include/libintl.h; \
```

图 9-20　安装 gftp 安装包

❺ 以 X-Windows 方式登录 Linux 系统，进入/usr/local/gftp/bin 目录，其中有个名为 gftp 的可执行程序。输入下面命令启动 gftp 程序，如图 9-21 所示。

```
./usr/local/gftp/bin/gftp
```

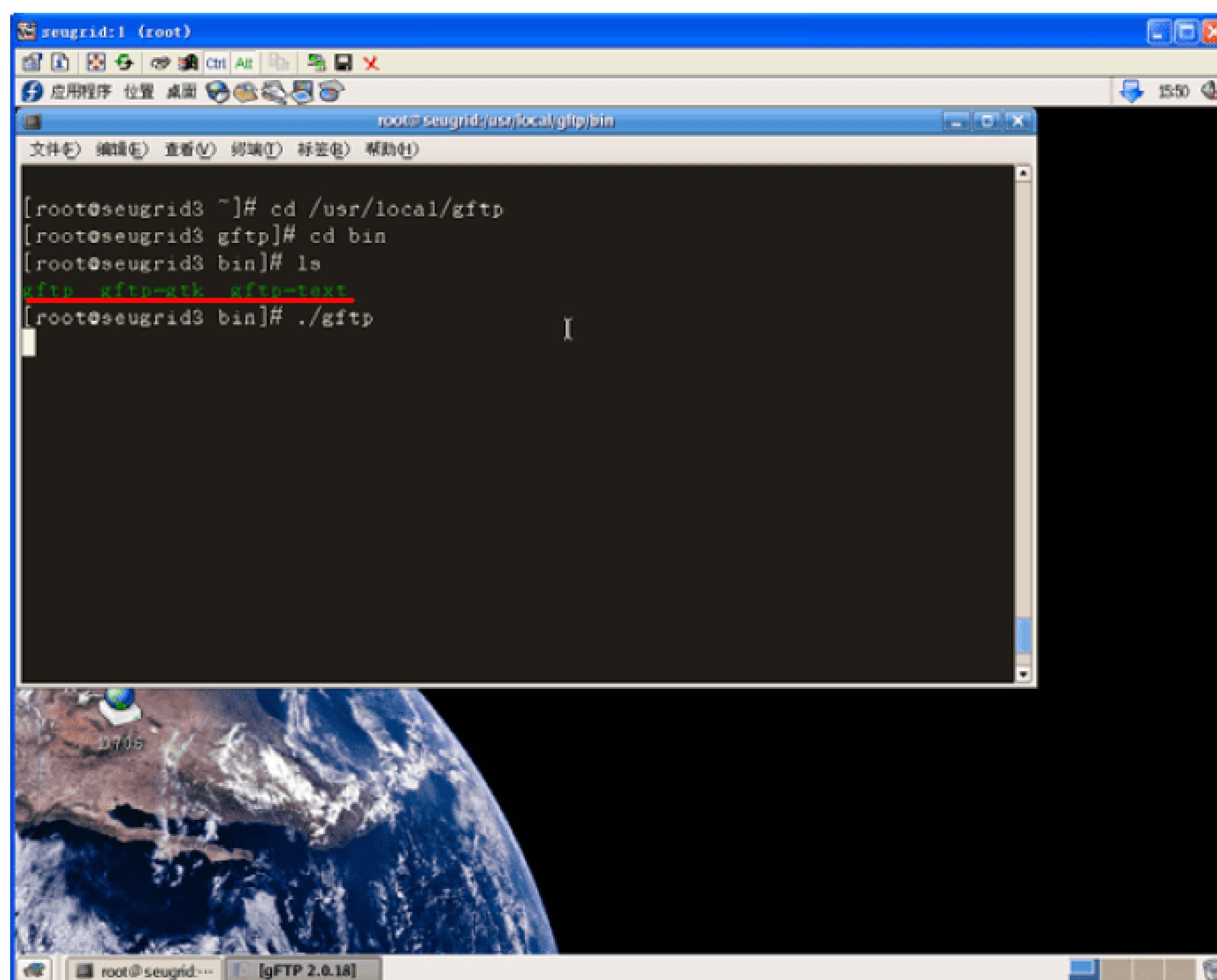


图 9-21　启动 gftp

❻ gftp 启动后，出现如图 9-22 所示的界面，正确输入 FTP 的 IP 地址，默认端口为 21，再输入用户名和密码，就能登录指定的 FTP。比如本例中是，ftptest:ftptest@localhost:21。界面左边窗口是本地 Linux 文件系统，右边窗口是 FTP 目录，利用中间的 ⇦ 和 ⇨ 按钮可以方便实现 FTP 的上传和下载功能。

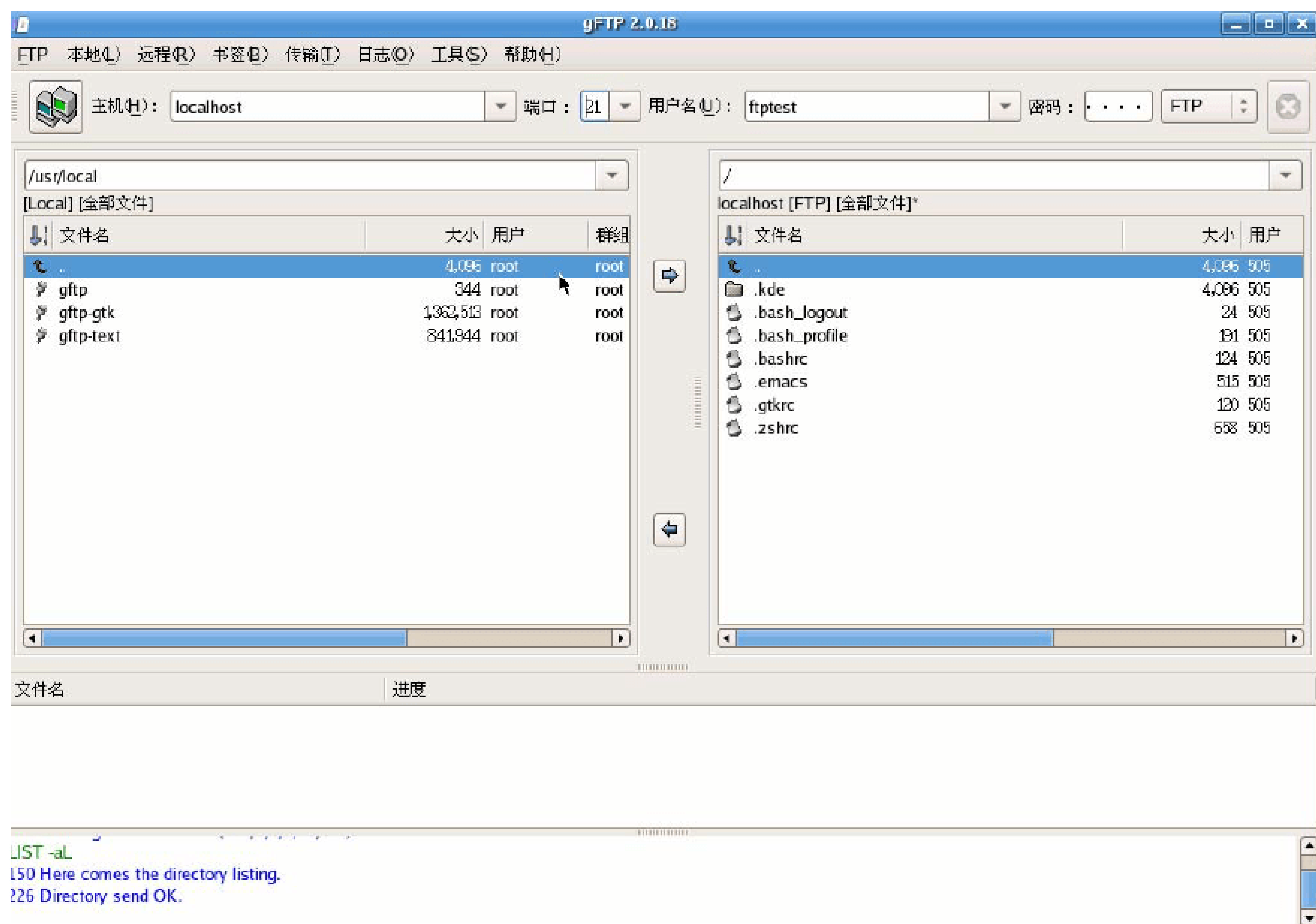


图 9-22　使用 gftp 登录本机 FTP

9.4　本 章 小 结

本章主要介绍了 FTP 服务的基本原理，使用 vsftpd 软件架设 FTP 服务器的方法，以及图形化界面软件 gftp 的安装和使用。读者需要重点掌握 FTP 的具体工作流程，注意区分主动模式和被动模式的应用场合。vsftd 作为一种常见的架设 FTP 服务器的软件，对其安装过程要熟练掌握，基本配置命令及参数意义也要非常熟悉。至于图形化界面 gftp 客户端程序的相关内容作一般了解即可，感兴趣的读者可通过其他参考资料作进一步的了解。本章内容相对比较简单， FTP 的相关应用读者们也大多非常熟悉，理解起来相信不会有多大困难。

第 10 章　电子邮件服务器的配置与架设

电子邮件服务是 Internet 上一项最基本的服务，也是最重要的服务之一。用户通过电子邮件可以方便快捷地与远程用户进行信息交互。目前 Internet 上有 30%以上的业务是电子邮件，仅次于万维网服务。本章主要介绍电子邮件服务的基本概念及工作原理，然后介绍自 Linux 诞生以来应用最广泛的邮件服务器软件 Sendmail 的配置，以及目前广泛流行、较 Sendmail 有较高安全性的邮件服务器软件 Postfix 的安装和配置。基于 Postfix 所架设的邮件转发代理，介绍如何使用 dovecot 实现 POP 和 IMAP 服务。最后简单介绍 Linux 系统下邮件客户端软件 Evolution。

通过本章的学习，读者应掌握以下内容：

- ✧ 电子邮件系统的工作原理
- ✧ Sendmail 的安装和配置
- ✧ Postfix 的安装和配置
- ✧ dovecot 的配置和启动
- ✧ Evolution 软件的配置和使用

10.1　电子邮件服务概述

10.1.1　电子邮件服务简介

电子邮件服务是 Internet 的基本服务之一，用于在 Internet 或 Intranet 上进行信息传递。与传统邮政的信件服务相比，电子邮件具有快速、高效、便捷的优点。传统邮件即使发份特快专递也需要一天时间，而发一份电子邮件通常只需要几分钟甚至以秒计就能到达对方，不管对方距离多远。电子邮件采用存储转发方式，发送邮件时无需接收方在线，收件人可以随时上网从邮件服务器上查阅邮件。

电子邮件服务是基于 Client/Server 模式的。一个完整的电子邮件系统通常主要包括以下几个部分：

- ✧ 邮件服务器，电子邮件系统的核心。它的主要功能是发送和接收邮件，并通知发件人邮件传送情况。根据用途上的区别邮件服务器可以分为发送邮件服务器和接收邮件服务器。发送邮件服务器通常为 SMTP 服务器，接收邮件服务器通常为 POP3 服务器或 IMAP4 服务器。
- ✧ 用户代理(user agent)，用户和电子邮件系统间的接口。它主要负责将邮件发送到邮

件服务器以及从邮件服务器上接收邮件。通常情况下，用户代理程序运行在邮件客户端。

- 电子邮件服务协议。目前用于电子邮件服务的协议主要有 SMTP、POP3 和 IMAP4 协议。SMTP(简单邮件传输协议)属于 TCP/IP 协议族，是一组用于从源地址向目的地址传递邮件的协议，可以控制邮件中转方式。电子邮件通过 SMTP 协议指定的服务器就可以传送到收件人的服务器。POP3 协议即邮局协议第 3 个版本，是电子邮件第一个离线协议标准，它规定了如何将个人 PC 连接到邮件服务器及下载电子邮件。POP3 允许把邮件从服务器上存储到本地主机和删除保存在服务器上的邮件。IMAP4 协议即 Internet 信息访问协议第 4 个版本，是一个 Client/Server 型协议，用于从本地服务器上访问电子邮件。用户的电子邮件由服务器负责接收和保存，用户可以通过浏览邮件头决定是否要下载邮件。此外，用户还可以在服务器上创建、更改文件夹或邮箱以及删除、检索邮件。

10.1.2　电子邮件服务工作原理

电子邮件服务工作流程如图 10-1 所示，我们假定两个服务器的域名分别为 nupt.edu.cn 和 sina.com，用户注册名分别 wu 和 tang，电子邮件地址分别为 wu@nupt.edu.cn 和 tang@sina.com。具体工作步骤如下。

用户在 POP 服务器上登记注册，被网络管理员设为授权用户后，取得一个 POP 邮箱，并获得 POP 和 SMTP 服务器的地址信息。

用户 wu 向用户 tang发送电子邮件时，电子邮件首先从客户端发送至 nupt.edu.cn 的 SMTP 服务器。

nupt.edu.cn 的 SMTP 服务器根据目的电子邮件地址查询 sina.com 的 SMTP 服务器，且转发该邮件。

sina.com 的 SMTP 服务器收到转发的电子邮件并保存。

用户 tang 利用客户端登录到 sina.com 的 POP 服务器，从其上下载和浏览电子邮件。

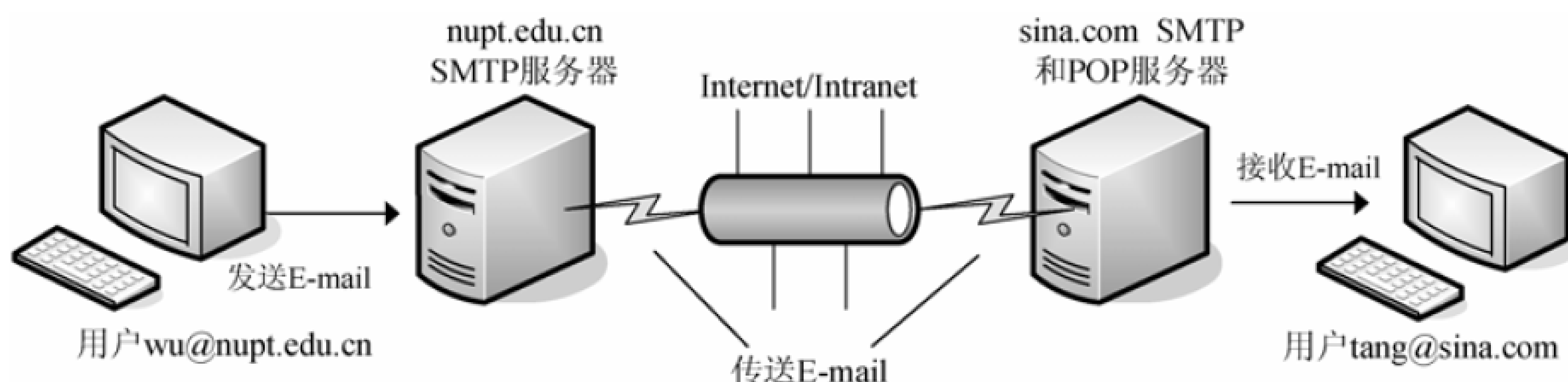


图 10-1　电子邮件传送流程

10.1.3 RELAY 与认证机制

MTA(Mail Transfer Agent，我们一般提到的邮件服务器就是 MTA)将信件传送到下一个 MTA 去的行为称为邮件转递(RELAY)。如果出现网络上所有的用户都可以借由这一个 MTA 进行 RELAY 的情况，则称之为 Open RELAY。当 MTA 由于设定不妥导致发生 Open RELAY，并且 MTA 此时是连上 Internet 的时候，由于 Internet 上使用 port scan 软件的人很多，该 MTA 具有 Open RELAY 功能的情况就会在短时间内被许多人察觉，此时那些不法广告、色情垃圾邮件等就会利用这个 Open RELAY MTA 进行发送，从而导致的问题主要有：

- ✧ MTA 主机所在的网域正常使用的联机速度将会变慢，因为网络频宽都被广告垃圾邮件占用了。
- ✧ MTA 主机可能由于大量发送信件导致主机资源被耗尽，容易产生不明原因的关机之类的问题。
- ✧ MTA 主机将会被 Internet 定义为黑名单，从此很多正常的邮件就会无法收发。
- ✧ MTA 主机所在的 IP 将会被上层 ISP 封锁，直到解决这个 Open RELAY 的问题为止。
- ✧ 如果该 MTA 主机被利用来发送黑客信件，则此 MTA 将会被追踪为最终站。

因此，一般都将 MTA 预设启动为仅监听内部循环接口，并且取消 Open RELAY 的功能。取消了 Open RELAY 功能后，用户必须取得合法使用该 MTA 的权限后才能利用这个 MTA 的 RELAY 功能来帮忙转信。通常设定 RELAY 的方法有以下两种。

(1) 规定某一个特定客户端的 IP 或网段，例如规定内部 LAN 的 202.119.1.0/24 可使用 RELAY。

(2) 如果客户端的 IP 不固定，则可以利用认证机制来处理。

上述的认证机制常见的有两种，即 SMTP 邮件认证机制和 SMTP after POP。这两种机制基本上都是让用户输入认证用的账号和密码来确定他是否有合法使用该MTA 的权限，然后对通过认证者开启 RELAY 的支持。这样，MTA 不需启动 Open RELAY，而客户端可以正常地利用认证机制来收发信件。

10.2 Sendmail 邮件服务的配置

应用实例导航——利用 Sendmail 架设 MTA 服务器

※场景呈现

某科研机构需要利用 Sendmail 软件架设邮件转发代理(Mail Transfer Agent，MTA)，该 MTA 服务需要在服务器的所有网卡上侦听，要求开放接收 cgsp.seu.edu.cn 和 seugrid2.seu.edu.cn 这两台主机的所有邮件，并对局域网内所有主机提供 RELAY 功能，局域

网段为 172.18.12.*。

※技术要领

(1) sendmail.mc 的设定。
(2) Sendmail 的启动。
(3) Sendmail 的 RELAY 功能权限设定。

Sendmail 是使用最广泛的简单邮件服务软件，几乎所有 Linux 系统都提供了这个软件，但是它的安全性较差。Sendmail 一般以 root 用户运行，所运行的进程权限很高，在 9.2 节"使用 vsftpd 架设 FTP 服务器"中提及一旦权限高的进程被黑客所利用，将给系统带来灾难性的后果；另外，Sendmail 只适合邮件数量较少的情况，无法满足高负载邮件系统的需要。

尽管如此，提到电子邮件服务器的架设，必须先谈一谈这款广泛应用的邮件服务软件。本节介绍如何使用 Sendmail 软件架设 MTA 服务，内容包括：Sendmail 软件所包括的一些组件简介；Sendmail 软件的配置方法；Sendmail 的 RELAY 权限设定。

10.2.1　Sendmail 软件结构简介

Fedora 6 默认安装 Sendmail 软件，因此不必考虑 Sendmail 的安装问题，它包含以下几个组件。

- ✧ sendmail：sendmail 服务的入口程序，在/etc/init.d 目录下。
- ✧ sendmail-cf：sendmail.cf 的设定文件。
- ✧ m4：将 sendmail-cf 这个设定文件转成实际可使用的设定数据。

几乎所有的 Sendmail 相关设定文件都在/etc/mail 目录下，下面对该目录所包含的几个主要设定文件作简单介绍。

- ✧ /etc/mail/sendmail.cf
  Sendmail 的主设定文件，不过，这个设定文件的内容是无法理解的，因此不能随意手动修改这个文件。那么怎样处理 sendmail.cf 设定文件呢？这就是 Sendmail 组件 sendmail-cf 和 m4 的作用，也就是说我们必须通过 sendmail-cf 和 m4 来修改 sendmail.cf 主设定文件，而不能手动修改。
- ✧ /usr/share/sendmail-cf/cf/*.mc
  所有*.mc 文件是 sendmail.cf 设定文件的输入数据，也就是说 m4 程序通过读入 sendmail-cf 中的*.mc 文件中的数据进行处理产生 sendmail.cf 文件。
- ✧ /etc/mail/sendmail.mc
  m4 程序还需要有 sendmail.mc 文件的配合来完成 sendmail.cf 设定文件的转化，读者可以通过图 10-2 来理解 m4 程序、sendmail.mc、sendmail-cf 和 sendmail.cf 之间的关系。我们可以看到，sendmail.mc 包含/usr/share/sendmail-cf/cf 的所有*.mc 文件，它作为 m4 程序的输入文件，处理后生成 sendmail.cf 文件。
- ✧ / etc/mail/local-host-names
  这个文件记录了能够接收邮件的所有信任主机的名字。

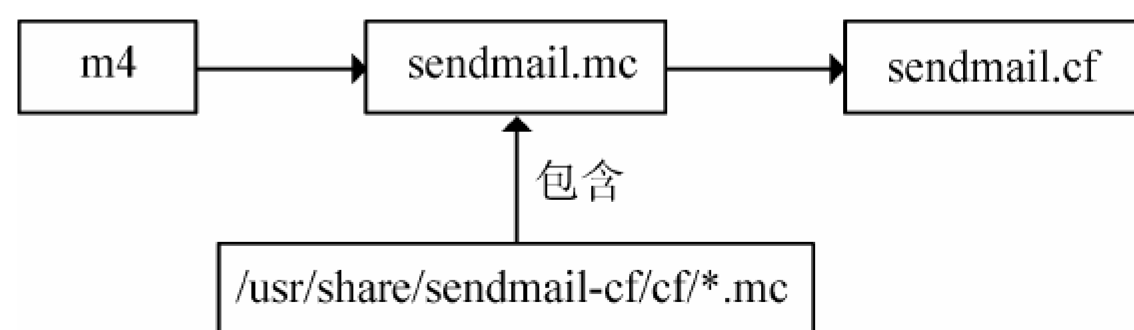


图 10-2　Sendmail 设定文件之间的关系

- ✧ /etc/mail/access.db
  设定 RELAY 主机权限的数据库文件，利用 makemap 程序对/etc/mail/access 设定文件的处理生成。
- ✧ /etc/aliases.db
  设定别名的数据库文件，利用 newaliases 程序对/etc/aliases 设定文件的处理生成。
- ✧ /var/spool/mqueue
  这个就是 Sendmail 的邮件队列，当邮件被 MTA 收下后，就会被放置到这里来等待 MDA 的处理。如果该封邮件是本机账号，那么就会被挪到/var/spool/mail/使用者账号去，然后从队列中移除该封邮件。如果该封邮件需要 RELAY，那么当邮件传送到下一部 MTA 后，信件数据就会从队列中移除。可以通过 mailq 命令查看当前队列的长度。

10.2.2　Sendmail 的配置与启动

有了对 Sendmail 软件的大概了解后，就可以进行 Sendmail 服务的配置和启动，具体步骤如下。

❶ Sendmail 服务在系统启动时已经启动，可以通过下面命令查看它的侦听状态，如图 10-3 所示。从图 10-3 的标注部分可以看到，Sendmail 初始状态只在本机环路网卡上侦听，也就是说不侦听有外部数据流量的网卡，因此这种状态无法接受和发送电子邮件，所以首先要将 sendmail 在所有网卡上都进行侦听。

```
netstat -tulnp | grep mail
```

```
[root@sec webmin-1.340]# netstat -tulnp | grep mail
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN      1992/sendmail: acce
[root@sec webmin-1.340]#
```

图 10-3　Sendmail 侦听环路网卡

❷ 使用 vi 命令打开 sendmail.mc 文件，如图 10-4 所示。找到 DAEMON_OPTIONS 这行语句，如图 10-5 所示，可以看到其中的 Addr 选项的默认设置为 127.0.0.1，这表示 Sendmail 服务只侦听本地环路网卡。我们将其改成：

```
DAEMON_OPTIONS('Port=smtp,Addr=0.0.0.1,Name=MTA')dnl
```

如图 10-6 所示，端口设定为 smtp，即简单邮件协议的端口，为 25 号端口；Addr 设定为 0.0.0.1，即侦听所有网卡；Name 为 MTA，即邮件转发代理类型。请注意，DAEMON_OPTIONS 的括号内，以 "`" 符号开头，以 "'" 即单引号结束。

```
[root@seugrid1 mail]# vi sendmail.mc
divert(-1)dnl
dnl #
dnl # This is the sendmail macro config file for m4. If you make changes to
dnl # /etc/mail/sendmail.mc, you will need to regenerate the
```

图 10-4　打开 sendmail.mc 文件

```
dnl # address restriction to accept email from the internet or intranet.
dnl #
DAEMON_OPTIONS(`Port=smtp,Addr=127.0.0.1, Name=MTA')dnl
dnl #
```

图 10-5　初始的 DAEMON_OPTIONS

```
dnl # address restriction to accept email from the internet or intranet.
dnl #
DAEMON_OPTIONS(`Port=smtp,Addr=0.0.0.0, Name=MTA')dnl
dnl #
```

图 10-6　修改后的 DAEMON_OPTIONS

❸ 使用 mv 命令备份原始的 sendmail.cf 文件，然后使用 m4 命令通过刚才设定好的 sendmail.mc 文件生成 sendmail.cf 文件，如图 10-7 所示。所用到的两条命令分列于下：

```
mv sendmail.cf sendmail.cf.bk
m4 sendmail.mc > sendmail.cf
```

```
[root@seugrid1 mail]# mv sendmail.cf sendmail.cf.bk
[root@seugrid1 mail]# m4 sendmail.mc > sendmail.cf
```

图 10-7　生成 sendmail.cf 文件

❹ 成功生成 sendmail.cf 文件后，需要重启 sendmail 服务使其生效，使用下面命令重启该服务，如图 10-8 所示，Sendmail 服务启动了两个进程。

```
/etc/init.d/sendmail restart
```

```
[root@seugrid1 mail]# /etc/init.d/sendmail restart
Shutting down sm-client:                                   [  OK  ]
Shutting down sendmail:                                    [  OK  ]
Starting sendmail:                                         [  OK  ]
Starting sm-client:                                        [  OK  ]
```

图 10-8　重启 Sendmail 服务

同理，启动、关闭 Sendmail 服务，查看 Sendmail 服务状态等命令格式如下：

```
/etc/init.d/sendmail  {stop|start|reload|condrestart|status}
```

如果再次使用 netstat 命令查看 Sendmail 服务的侦听状态，可以发现 Sendmail 服务已经在所有网卡上进行侦听了，如图 10-9 标注部分所示。

```
[root@seugrid1 mail]# netstat -tulnp | grep mail
tcp        0      0 0.0.0.1:25              0.0.0.0:*               LISTEN      19984/sendmail: acc
[root@seugrid1 mail]#
```

图 10-9　Sendmail 服务侦听所有网卡

❺ 按照场景呈现中的要求，我们需要开放接收 cgsp.seu.edu.cn 和 seugrid2.seu.edu.cn 这两台主机的所有邮件，只需将这两台主机名添加到/etc/mail/local-host-names 文件中即可，如图 10-10 所示。仍然需要重启 Sendmail 服务使 local-host-names 文件的设定生效，重启方法见步骤(4)。

```
[root@seugrid1 mail]# vi local-host-names
# local-host-names - include all aliases for your machine here.

cgsp.seu.edu.cn
seugrid2.seu.edu.cn
```

图 10-10　/etc/mail/local-host-names 文件的设置

❻ 场景呈现还要求对局域网段 172.18.12.*内所有主机提供 RELAY 功能，需要对/etc/mail/access 设定文件进行设置，在该文件添加如下语句，如图 10-11 所示。

```
172.18.12    RELAY
```

```
[root@seugrid1 mail]# vi access
# Check the /usr/share/doc/sendmail/README.cf file for a description
# of the format of this file. (search for access_db in that file)
# The /usr/share/doc/sendmail/README.cf is part of the sendmail-doc
# package.
#
# by default we allow relaying from localhost...
Connect:localhost.localdomain           RELAY
Connect:localhost                       RELAY
Connect:127.0.0.1                       RELAY

172.18.12                               RELAY
```

图 10-11　/etc/mail/access 文件的设置

可以看到初始的 access 文件规定了本机的邮件才能 RELAY，这实际上是基于安全的考虑，在第 10.2.1 节中我们讲述过对所有主机邮件都 RELAY 将带来许多安全上的问题，会给网络黑客提供攻击其他服务器的中介。

对/etc/mail/access 文件设定完毕后，进入/etc/mail 目录，使用下面命令使刚才的 access 文件设定生效，如图 10-12 所示。

```
makemap hash access < access
```

```
[root@seugrid1 mail]# pwd
/etc/mail
[root@seugrid1 mail]# makemap hash access < access
[root@seugrid1 mail]#
```

图 10-12　makemap 使 access 设置生效

点评与拓展：写入/etc/mail/local-host-names 文件的主机名，必须是由 DNS 正向解析能得出的合法主机名。

10.3　Postfix 邮件服务的配置

应用实例导航——利用 Postfix 架设 MTA 服务器

※场景呈现

由于某科研机构用户的增多，用 Sendmail 软件架设的 MTA 已不能满足需要。该机构准备利用 Postfix 软件重新架设 MTA，实现与 Sendmail 所架设的 MTA 类似的功能：MTA 服务在服务器的所有网卡上侦听，开放接收 cgsp.seu.edu.cn 和 seugrid2.seu.edu.cn 这两台主机的所有邮件，并对局域网内所有主机提供 RELAY 功能，局域网段为 172.18.12.*。

※技术要领

(1) 熟悉 Postfix 软件的组成。
(2) Postfix 的安装和启动。
(3) Postfix 主配置文件的意义和设定。

Postfix 是由 IBM 资助，Wietse Zweitze Venema 博士负责开发的邮件服务软件。Venema 博士在 1998 年首次发布这个自行开发的邮件服务器，并定名为 VMailer。不过，VMailer 这个名字与其他已注册的商标很类似，为了避免法律纠纷，Venema 博士将这个邮件服务软件的名称改为 Postfix。

Postfix 在设计理念方面，试图完全兼容 Sendmail 同时又要内核新颖，因此 Postfix 改进了 Sendmail 安全性上的问题，及 mail server 的工作效率，且使得设定文件更加简易。

本节首先介绍 Postfix 软件各组成部分的功能，然后讲述 Postfix 的安装与启动，重点讲述 Postfix 主配置文件的意义和设定方法。

10.3.1　Postfix 软件结构简介

Postfix 的设定文件都在/etc/postfix/目录中，下面对该目录所包含的几个主要设定文件作简单介绍：

✧ /etc/postfix/main.cf
postfix 的主设定文件，只要修改过这个档案，就要重新启动 postfix 使其生效。

✧ /etc/postfix/master.cf
规定了 postfix 每个程序的运行参数，一般无需改动此设定文件。

✧ /etc/postfix/access
与 sendmail 的/etc/mail/access 具有类似的功能，用于设定 RELAY 主机权限的数据库文件，利用 makemap 程序对/etc/mail/access 设定文件的处理生成。

✧ /etc/aliases (利用 s 均可)
用于设定使用者别名的文件，利用 postalias 或 newaliase 程序将 aliases 转化成 aliases.db 数据库文件。

postfix 的主要执行程序如下。

✧ /usr/sbin/postconf
该命令可以列出 postfix 的详细设定数据，包括系统默认值也会被列出来；如果仅需列出非默认值的设定数据，则使用“postconf –n”命令。

✧ /usr/sbin/postfix (主要的 daemon 指令)
启动 postfix 的入口程序，启动这个程序会重新读取设定文件。

✧ /usr/sbin/postalias
设定使用者别名数据库的命令，将 aliases 转化成 aliases.db 数据库文件，即将 ASCII 格式的文件转换为数据库。

✧ /usr/sbin/postcat
该命令用于检查队列中的邮件内容，即查看目录/var/spool/postfix 中的内容。

✧ /usr/sbin/postmap
该命令用于/etc/mail/access 设定文件。

介绍完 postfix 的设定文件和主要执行程序以后，就可以安装 postfix 软件了。

❶ Fedora 6 安装盘提供了 Postfix 的 rpm 安装包，但不默认安装。在安装 Postfix 之前，必须停止 Sendmail 服务。输入下面命令停止 Sendmail 服务，并使用 chkconfig 命令关闭 Sendmail 开机自启动功能，如图 10-13 所示，所用到的三条命令分列于下:

```
/etc/init.d/sendmail stop
chkconfig sendmail off
chkconfig sendmail --list
```

```
[root@seugrid1 mail]# /etc/init.d/sendmail stop
Shutting down sm-client:                                   [  OK  ]
Shutting down sendmail:                                    [  OK  ]
[root@seugrid1 mail]# chkconfig sendmail off
[root@seugrid1 mail]# chkconfig sendmail --list
sendmail        0:off   1:off   2:off   3:off   4:off   5:off   6:off
```

图 10-13 停止 Sendmail 服务

❷ Postfix 的 rpm 安装包的名字为 postfix-2.3.3-2.i386.rpm，使用下面命令安装 Postfix 软件，如图 10-14 所示。

```
rpm -ivh postfix-2.3.3-2.i386.rpm --force -nodeps
```

```
[root@seugrid1 local]# rpm -ivh postfix-2.3.3-2.i386.rpm --force --nodeps
warning: postfix-2.3.3-2.i386.rpm: Header V3 DSA signature: NOKEY, key ID 4f2a6fd2
Preparing...                ########################################### [100%]
   1:postfix                ########################################### [100%]
```

图 10-14　安装 Postfix 软件

安装成功后，即可使用下面命令启动 Postfix 服务，如图 10-15 所示。

```
/etc/init.d/postfix start
```

```
[root@seugrid1 local]# /etc/init.d/postfix start
Starting postfix:                                          [  OK  ]
```

图 10-15　启动 Postfix 服务

10.3.2　Postfix 的配置和启动

Postfix 软件的配置目标与 Sendmail 类似，也是将其设定为在所有网卡上都侦听数据，并设定接收邮件的主机和网域，以及设定提供 RELAY 功能的主机范围等。与 Sendmail 不同的是，Postfix 软件的设定都在/etc/postfix/main.cf 文件内。下面讲述如何配置 main.cf 这个文件。

❶ 打开/etc/postfix/main.cf 文件，该文件尽管内容很多，但其中大部分内容都是注释，由“#”符号开头，我们只需在注释的基础上作少量的修改即可。

第一处重要设置是主机名，后面很多设置都直接调用这个变量，找到 myhostname 这一行，如图 10-16 标注部分所示，根据具体情况输入你的主机名。

```
# The myhostname parameter specifies the internet hostname of this
# mail system. The default is to use the fully-qualified domain name
# from gethostname(). $myhostname is used as a default value for many
# other configuration parameters.
#
#myhostname = host.domain.tld
myhostname = seugrid1.seu.edu.cn
```

图 10-16　设定主机名

接着，找到 myorigin 这一行，将上面设定的主机名作为 myorigin 的值，如图 10-17 所示。myorigin 参数记录由本台主机发出的邮件的邮件头中的 mail from 地址，即电子邮件源主机的主机名。

```
# For the sake of consistency between sender and recipient addresses,
# myorigin also specifies the default domain name that is appended
# to recipient addresses that have no @domain part.
#
myorigin = $myhostname
#myorigin = $mydomain
```

图 10-17　设定发信源主机名

❷ Postfix 默认情况下只侦听本机环路网卡(lo 127.0.0.1)，找到 inet_interfaces 这一行，将此参数设为 all，这样 Postfix 才能在主机的所有网卡上都进行侦听，如图 10-18 所示。inet_interfaces 的设定相当于 Sendmail 的 DAEMON_OPTIONS 设定项。

```
# Note: you need to stop/start Postfix when this parameter changes
#
inet_interfaces = all
#inet_interfaces = $myhostname
#inet_interfaces = $myhostname, localhost
#inet_interfaces =
```

图 10-18 inet_interfaces 设定

❸ 接着设定 Postfix 允许接收邮件的主机名，如同 Sendmail 中的 local-host-names 文件的设置一样，找到 mydestination 这一行，默认允许接收本机和本域的邮件，加入额外允许接收邮件的主机名，本例中为 cgsp.seu.edu.cn 和 seugrid2.seu.edu.cn，如图 10-19 标注部分所示。习惯上，将新添的主机名另起一行，不同主机名之间用逗号隔开。也可以将 mydestination 参数设为与 Sendmail 类似的 local-host-names 文件，格式如下:

```
mydestination = /etc/postfix/local-host-names
```

然后同样在 local-host-names 中添加允许接收邮件的主机名，这样的设置方法体现出 postfix 对 sendmail 的兼容。

```
# See also below, section "REJECTING MAIL FOR UNKNOWN LOCAL USERS".
#
mydestination = $myhostname, localhost.$mydomain, localhost,
                 cgsp.seu.edu.cn, seugrid2.seu.edu.cn
#mydestination = $myhostname, localhost.$mydomain, localhost, $mydomain
#mydestination = $myhostname, localhost.$mydomain, localhost, $mydomain,
#       mail.$mydomain, www.$mydomain, ftp.$mydomain
```

图 10-19 mydestination 设定

❹ postfix 的 main.cf 文件中还有一项为 mynetworks_style，它设定了信任网域，即可以提供 RELAY 功能的网域。但是由于 mynetworks 设定项提供了同样的功能，因此一般只设定 mynetworks 选项，对于 mynetworks_style 项，找到 mynetworks 这一行，设定如图 10-20 标注部分所示的内容。

```
# You can also specify the absolute pathname of a pattern file instead
# of listing the patterns here. Specify type:table for table-based lookups
# (the value on the table right-hand side is not used).
#
mynetworks =  127.0.0.0/8,hash:/etc/postfix/access
#mynetworks = $config_directory/mynetworks
#mynetworks = hash:/etc/postfix/network_table
```

图 10-20 信任网域的设定

其中，127.0.0.0/8 表示对本机环路网域提供 RELAY 功能，其他信任网域的设定放在/etc/postfis/access 文件中。下节将详细讲述 access 的设置和生效方法。

❺ 实际上 mynetworks 设定的是信任的上游主机，即对 mynetworks 中发来的邮件提供 RELAY 功能，那么将这些邮件 RELAY 到何方呢？即下游主机在哪里呢？RELAY_domains 项就设定了下游主机的范围，一般设为 mydestination 变量，如图 10-21 所示。

```
# NOTE: Postfix will not automatically forward mail for domains that
# list this system as their primary or backup MX host. See the
# permit_mx_backup restriction description in postconf(5).
#
relay_domains = $mydestination
```

图 10-21　RELAY_domains 的设定

❻ 最后需要进行与别名相关的设定，这里只介绍主文件的别名设定，有关别名的用途和其他设定将在下节中讲述。找到 alias_maps 这一行，将其指向/etc/aliases 文件，再将 alias_database 项也同样指向/etc/aliases 文件，如图 10-22 标注部分所示。

```
#alias_maps = dbm:/etc/aliases
alias_maps = hash:/etc/aliases
#alias_maps = hash:/etc/aliases, nis:mail.aliases
#alias_maps = netinfo:/aliases

# The alias_database parameter specifies the alias database(s) that
# are built with "newaliases" or "sendmail -bi".  This is a separate
# configuration parameter, because alias_maps (see above) may specify
# tables that are not necessarily all under control by Postfix.
#
#alias_database = dbm:/etc/aliases
#alias_database = dbm:/etc/mail/aliases
alias_database = hash:/etc/aliases
#alias_database = hash:/etc/aliases, hash:/opt/majordomo/aliases
```

图 10-22　别名的设定

❼ 这样 Postfix 的 main.cf 文件终于设定完了，使用 postmap 和 postalias 处理一下 access 和 aliases 文件。然后，利用 Postfix 命令检查主配置文件是否有语法错误。如果语法无误，就重启 Postfix 服务使 main.cf 配置生效，如图 10-23 所示，使用到的命令分列于下:

```
postmap hash:/etc/postfix/access
postalias hash:/etc/aliases
postfix check
/etc/init.d/postfix restart
```

使用下面命令查看 Postfix 服务的侦听状态，如图 10-24 所示，可以看到 Postfix 已经在所有网卡上侦听了，所在的仍然是 smtp 所规定的 25 号端口。

```
netstat -tulnp | grep ':25'
```

```
[root@seugrid1 postfix]# postmap hash:/etc/postfix/access
[root@seugrid1 postfix]# postalias hash:/etc/aliases
[root@seugrid1 postfix]# postfix check
[root@seugrid1 postfix]# /etc/init.d/postfix restart
Shutting down postfix:                                     [  OK  ]
Starting postfix:                                          [  OK  ]
```

图 10-23　重启 Postfix 服务使设置生效

```
[root@seugrid1 postfix]# netstat -tulnp | grep ':25'
tcp        0      0 0.0.0.0:25                  0.0.0.0:*                   LISTEN      29201/master
You have mail in /var/spool/mail/root
[root@seugrid1 postfix]#
```

图 10-24　查看 Postfix 侦听状态

点评与拓展：main.cf 文件中的变量跟 shell 中的变量格式一样，都是以“$”符号开头。若在 main.cf 文件中重复设定某一变量，以最后一次的设定为准。

10.3.3　Postfix 的其他配置

应用实例导航——Postfix 的主机过滤和虚拟别名

※场景呈现

该科研机构在利用 Postfix 软件架设 MTA 实现了基本功能后，需要进一步利用 access 文件实现主机过滤，要求接收所有来自 172.18.12.*网段的邮件，并拒绝来自.com 结尾的地址的商业邮件。

另外，需要使用别名机制实现群组邮递功能，使用 tian 用户接管 root 用户的邮件，并实现利用 teacher 用户管理学生群组的邮件的投递功能。

※技术要领

(1) 利用 access 文件实现主机过滤功能。

(2) 利用 aliases 文件实现虚拟别名功能。

前面讲述 main.cf 文件的设定时，遗留了两个问题，一个是 postfix 是如何使用 access 文件进行信任主机过滤的；另一个是别名文件 aliases 有何意义，且是如何设定的。本节详细解释这两个问题。

1．access 信任主机过滤

信任主机的过滤规则类似于 Linux 防火墙中的规则，分为接受(ACCEPT)和拒绝(REJECT)两种。/etc/postfix/access 用于设定邮件主机使用权限与过滤机制，如果 mynetworks

项指定到 access 文件，我们就需要对 access 进行设定。基本的 access 语法如下：

规范的范围或规则　　　　　　　Postfix 的动作

```
IP/部分 IP/主机名/Email 等      ACCEPT/REJECT
```

根据 ACCESS 的语法，场景呈现中要求开放 172.18.12.*网段的权限，那么只要在 access 中写入：

```
172.18.12                       ACCEPT
```

如果要过滤掉所有.com 结尾的地址发来的邮件，只要在 access 中写入：

```
.com                            REJECT
```

access 文件设定如图 10-25 所示。然后使用下面命令使 access 文件生效，如图 10-26 所示。

```
postmap hash:/etc/postfix/access
```

```
[root@seugrid1 ~]# vi /etc/postfix/access
172.18.12                          ACCEPT
.com                               REJECT
```

图 10-25　access 文件

```
[root@seugrid1 ~]# postmap hash:/etc/postfix/access
[root@seugrid1 ~]#
```

图 10-26　使 access 文件生效

2．aliases 别名文件设置

aliases 别名又称为虚拟别名功能，它能建立虚拟别名用户与真实系统用户之间的映射，即能将发送到虚拟别名用户的邮件投递到真实用户(或用户组)的邮箱中。用户别名机制是通过别名表在系统范围内实现别名邮件地址到真实用户邮件地址的重定向，别名表就是主设定文件中 alias_maps 和 alias_database 两个设定项所指向的文件，默认为/etc/aliases。下面详细说明/etc/aliases 的设置方法。

❶ 打开/etc/aliases 文件，如图 10-27 所示，可以看到/etc/aliases 文件以如下格式来组织：

```
[虚拟别名]:                  [真实用户]
```

如果我们添加 “gao:　tang” 这条语句，即用 tang 这个用户接管了 gao 这个用户的邮箱，所有发到 gao 用户的邮件，全部被重定向到 tang 这个用户中。如果需要通过 tian 这个用户来查看 root 用户的邮件，只要添加“root:　root,tian”语句，表示所有发送到 root 用户的邮件将被转交到 root 和 tian 两个用户的邮箱中，如图 10-28 所示。假设添加的语句是：“root:　tian”，那么从此 root 用户将再也接收不到自己的邮件了，这些邮件都转交到了 tian 用户的邮箱中。

```
# Basic system aliases -- these MUST be present.
mailer-daemon:  postmaster
postmaster:     root

# General redirections for pseudo accounts.
bin:            root
daemon:         root
adm:            root
lp:             root
sync:           root
shutdown:       root
```

图 10-27　/etc/aliases 文件

```
# General redirections for pseudo accounts.

gao:                tang
root:               root,tian
```

图 10-28　添加别名

❷ postfix 的别名的一个实用功能就是实现群组邮递。假设一位教师需要为所有的学生发送同样的邮件，如果一个一个地发送将耗费大量的时间和精力，这时我们就可以用一个虚拟用户作为所有学生账号的别名，这样教师只需将邮件发送到这个虚拟账号中去就可被所有的学生接收了。群组邮递功能很实用，可以大幅度地提高工作效率，而且设定很简单，只要在/etc/ aliases 中添加下面的一行语句:

```
teacher:  st001,st002,st003, …
```

有多少学生都可以添加在这条语句之后，中间用逗号分隔，如图 10-29 所示。

```
# General redirections for pseudo accounts.

teacher:          st001,st002,st003
```

图 10-29　群组邮递

❸ 修改/etc/ aliases 之后，使用下面命令使设置生效，如图 10-30 所示。

```
postalias hash:/etc/aliases
```

```
[root@seugrid1 postfix]# postalias hash:/etc/aliases
[root@seugrid1 postfix]#
```

图 10-30　使/etc/aliases 生效

点评与拓展： 在讲述 aliases 的几个实例中，可以看到/etc/aliases 文件中左侧的虚拟别名既可以是实际不存在的用户名，如 teacher 用户；也可以是实际存在的系统用户，如 root、gao 用户。

10.4　POP 和 IMAP 服务

应用实例导航——利用 dovecot 架设 POP 服务器

※场景呈现

该科研机构在 MTA 提供 SMTP 功能服务的基础上，需要利用 POP 或 IMAP 服务提供邮件的转发和本地的分发功能，实现完整的电子邮件服务器的功能。要求使用 dovecot 服务实现 POP 服务器功能。

※技术要领

(1) dovecot 的安装。
(2) dovecot 的配置和启动。

Postfix 只是一个 MTA，只提供 SMTP 服务，即只负责邮件的转发及本地分发。要实现邮件异地接收，还必须安装 POP 或 IMAP 服务。通常都是将 STMP 和 POP 或 IMAP 服务安装在同一台主机上，这台主机即称为电子邮件服务器。dovecot 就是可以同时提供 POP 和 IMAP 服务的软件。

❶ 在安装 dovecot 服务前，可以先使用下列命令检查系统是否已经安装了 dovecot。

```
rpm -q dovecot
```

系统当前如果还没有安装 dovecot 服务，可以使用下列命令进行安装：

```
rpm -ivh dovecot-1.0-0.1.rc7.fc6.i386.rpm
```

命令执行结果如图 10-31 所示。

```
[root@seugrid2 local]# rpm -ivh dovecot-1.0-0.1.rc7.fc6.i386.rpm
warning: dovecot-1.0-0.1.rc7.fc6.i386.rpm: Header V3 DSA signature: NOKEY, key ID 4f2a6fd2
Preparing...                ########################################### [100%]
        package dovecot-1.0-0.1.rc7.fc6 is already installed
[root@seugrid2 local]#
```

图 10-31　dovecot 的安装

❷ dovecot 服务的配置文件是/etc/dovecot.conf，要启用最基本的 dovecot 服务，只要用下列命令修改配置文件中的相关内容即可。

```
protocols = pop3
listen = *
```

第一条语句用于指定电子邮件服务主机所运行的服务协议，第二条语句用于指定 POP3 服务所监听的网络接口，“*”表示监听本机上所有网络接口。

命令执行结果如图 10-32 及图 10-33 所示。

```
# Protocols we want to be serving: imap imaps pop3 pop3s
protocols =  pop3
```

图 10-32　指定服务协议

```
#      listen = *:10100
#      ..
#    }
listen = *
```

图 10-33　指定监听的网络接口

❸ 启动 dovecot 服务的命令如下，关闭、重启 named 服务，查看 named 服务状态等命令格式亦列于下方。

```
/etc/init.d/dovecot start
/etc/init.d/dovecot  {stop|restart|reload|condrestart|status}
```

启动命令执行结果如图 10-34 所示。dovecot 和 Postfix 服务安装配置完毕后，电子邮件客户端就可以利用该电子邮件服务器收发邮件了。

```
[root@seugrid2 local]# /etc/init.d/dovecot start
Starting Dovecot Imap:                                     [  OK  ]
```

图 10-34　启动 dovecot 服务

10.5　电子邮件客户端的配置

电子邮件客户端软件有很多且各有特色，但不管是在什么样的操作系统平台上运行，各个客户端软件配置步骤及需要设置的参数基本相同。Evolution 是一个以邮件处理为中心，集日程安排、任务管理等功能于一身的电子邮件客户端软件。下面就以 Evolution 为例来详细说明客户端的配置。

❶ 单击桌面上的电子邮件图标，如果是第一次运行系统则会打开如图 10-35 所示的欢迎界面。

单击【前进】按钮，打开【标识】设置界面，分别输入用户名和电子邮件地址，并选中【使它成为我的默认客户】复选框，具体如图 10-36 所示。

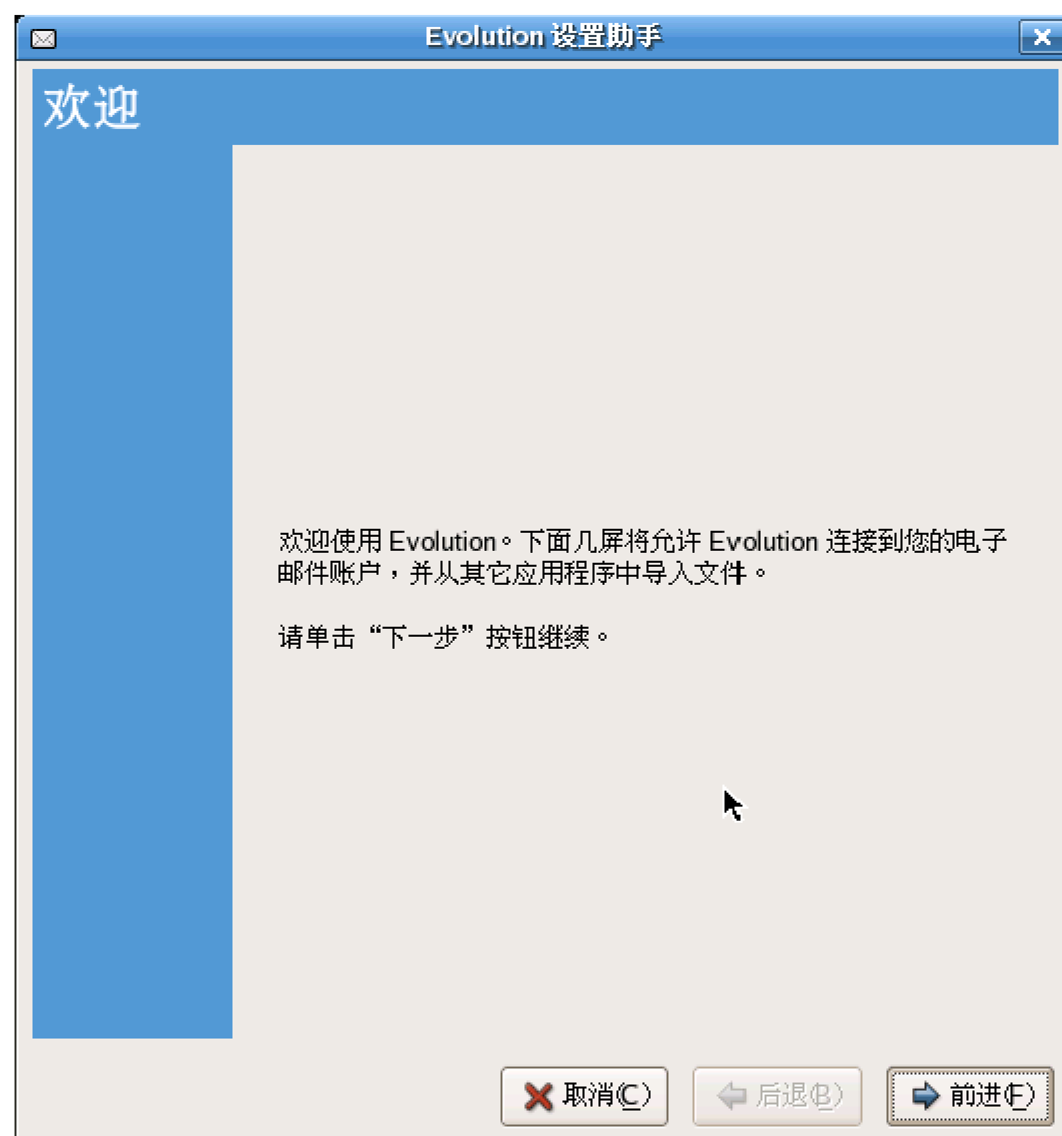


图 10-35　【欢迎】界面

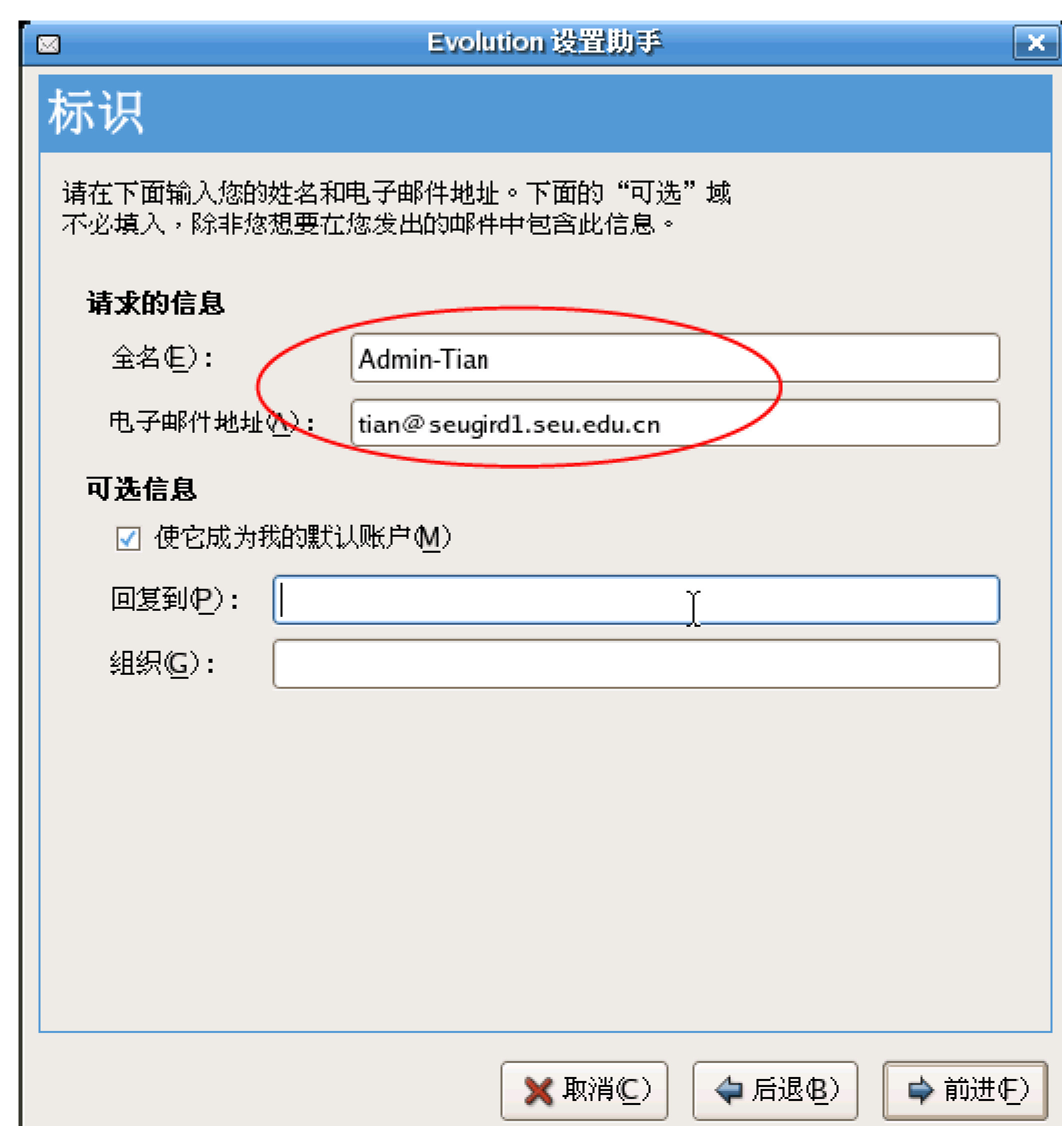


图 10-36　【标识】设置界面

❷ 单击【前进】按钮，打开【接收电子邮件】设置界面，如图 10-37 所示。在【服务器类型】下拉列表框中选择接收邮件服务器类型，这里我们选择 POP。在【服务器】文本框中输入接收邮件服务器主机的域名或 IP 地址，这里输入 seugrid1.seu.edu.cn。在【用户名】文本框

中输入用户账户名，这里输入 tian。单击【检查支持的类型】按钮，先让客户端自动测试该邮件服务器支持的认证类型，再从【认证类型】下拉列表框中选择验证用户密码的方式，这里选择【密码】方式。

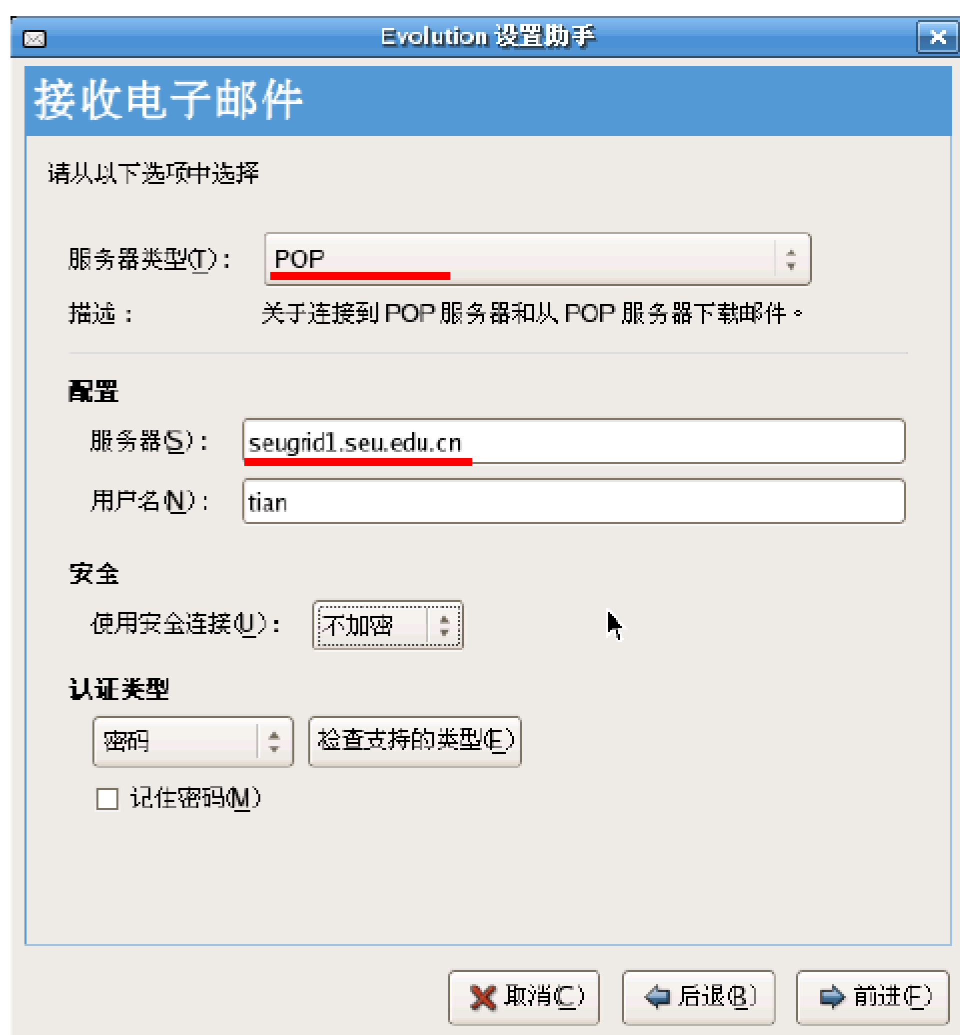


图 10-37 【接受电子邮件】设置界面

❸ 单击【前进】按钮，打开【接收选项】设置界面，选中【自动检查新邮件的间隔】复选框，再根据个人需要设置其间隔时间，我们这里选“10”，如图 10-38 所示。【信件存储】选项组中的复选框可以根据用户需要自行选择。

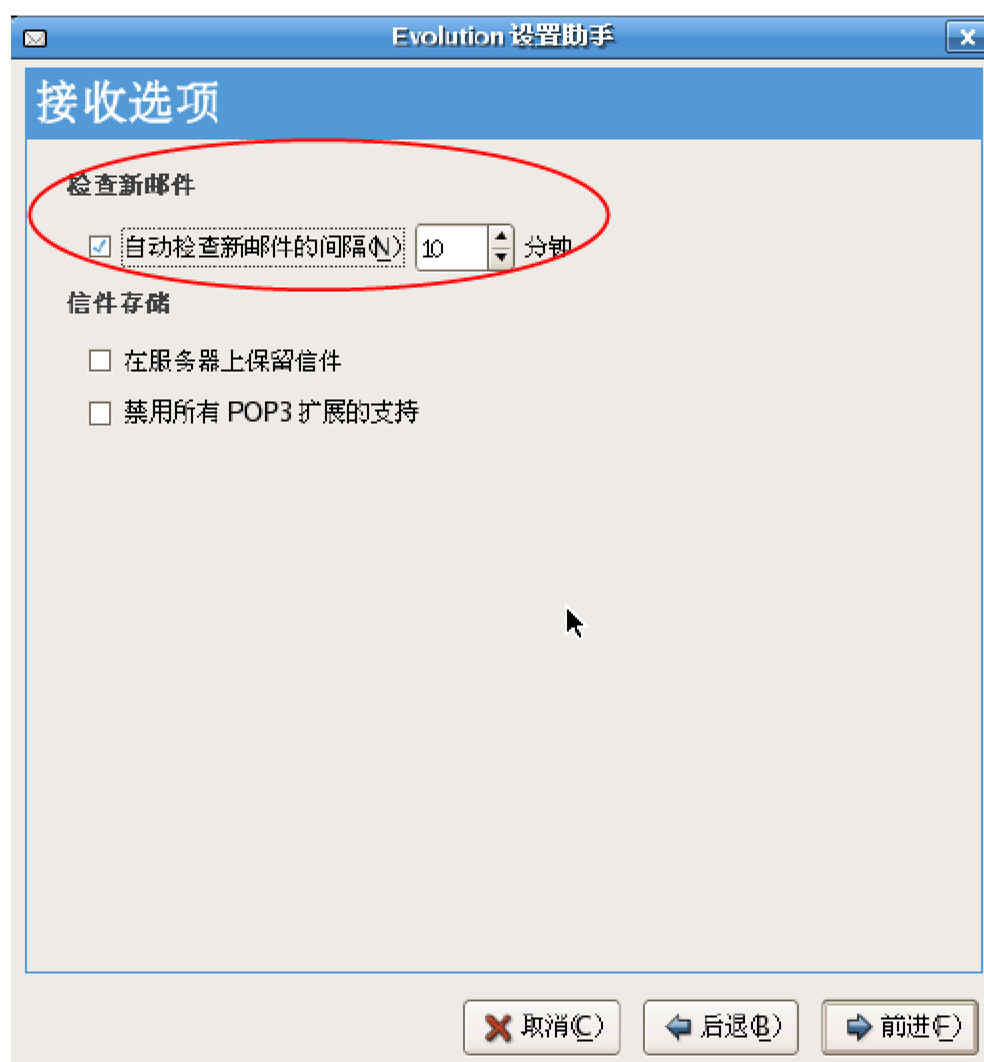


图 10-38 【接收选项】设置界面

❹ 单击【前进】按钮，打开【发送电子邮件】设置界面，如图 10-39 所示，在该设置界面中

进行如下设置:

在【服务器类型】下拉列表中选择发送邮件服务器的类型，这里我们选择 SMTP。在【服务器】文本框中输入发送邮件服务器的域名或 IP 地址，这里输入 seugrid1.seu.edu.cn。如果发送邮件服务器启用了认证功能，则选中【服务器需要认证】复选框。单击【检查支持的类型】按钮，先让客户端自动测试邮件服务器所支持的认证类型，再从【类型】下拉列表框中选择密码认证方式，这里我们选中 Login。在【用户名】文本框中输入用户账户名 tian。此处设置类似于 POP 服务器的设置。

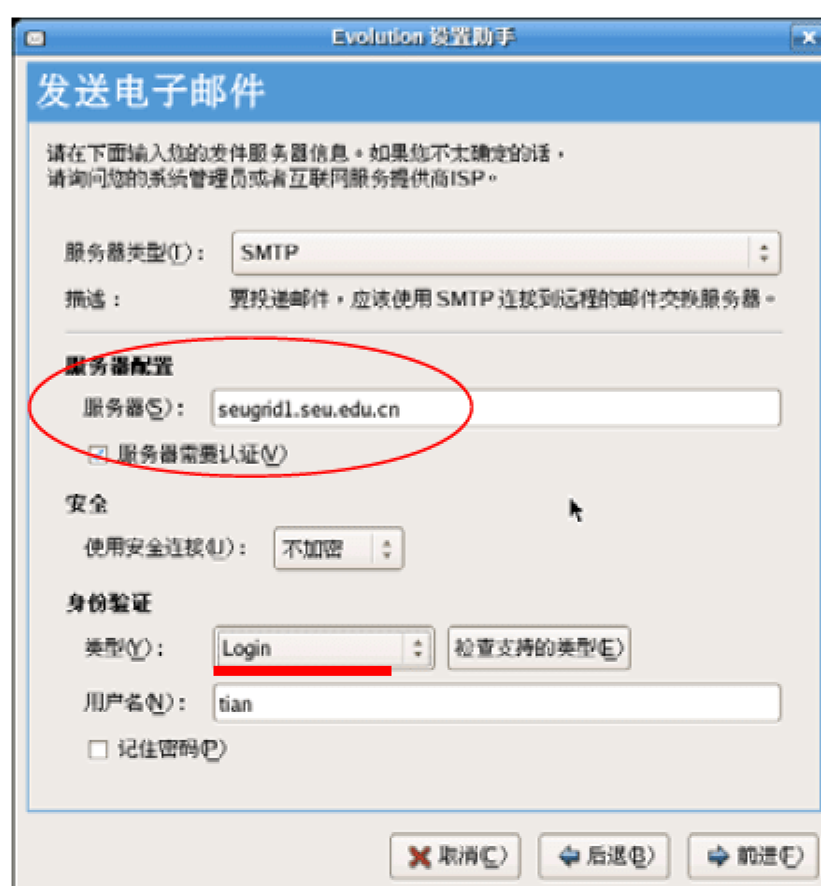


图 10-39　【发送电子邮件】设置界面

❺ 单击【前进】按钮，打开【账户管理】窗口，在【名称】文本框中输入描述性名称，以便于账户管理。再继续单击【前进】按钮，打开 Timezone 设置界面，在此处进行时区设置，这里选择“亚洲/上海”，如图 10-40 所示。



图 10-40　时区设置

❻ 单击【前进】按钮，再单击【应用】按钮，保存设置后即可打开 Evolution 工作窗口，如图 10-41 所示。这样我们就完成了整个初始配置过程。初始化配置完成后，就可以在 Evolution 工作窗口进行收发邮件及管理邮件等工作。工作过程中可以随时根据需要改变前面所作的设置。

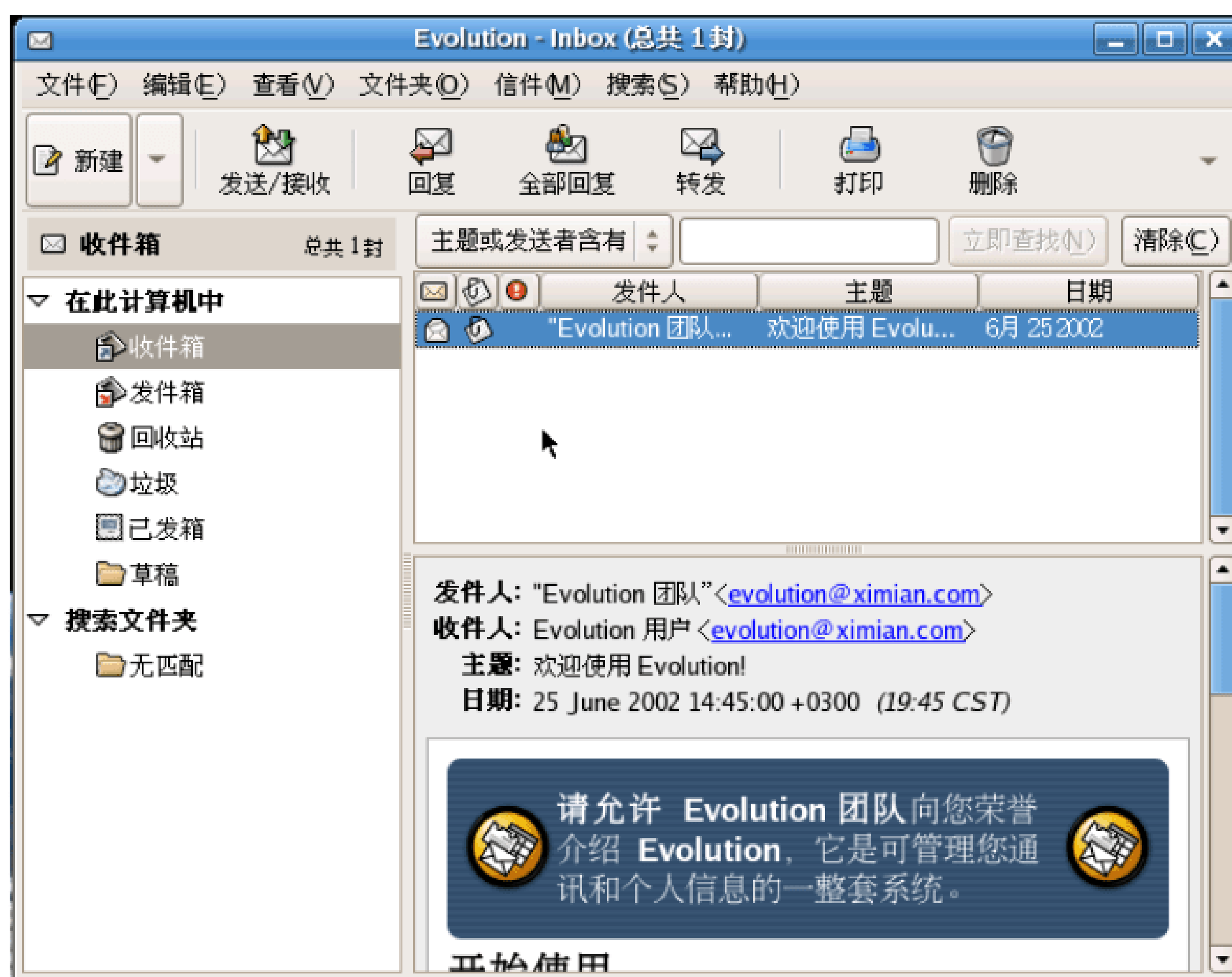


图 10-41 Evolution 工作窗口

📖**点评与拓展：** Evolution 客户端软件类似于 Windows 下面的 Outlook 和 Foxmail 等，从配置方法来讲更类似于 Outlook 软件。

10.6 本 章 小 结

本章主要介绍电子邮件服务的基本概念及工作原理，然后介绍自 Linux 诞生以来应用最广泛的邮件服务器软件 Sendmail 的配置，以及目前广泛流行、较 Sendmail 有较高安全性的邮件服务器软件 Postfix 的安装和配置。在介绍这两种软件配置的同时，还穿插介绍了两种软件的发展及其区别。基于 Postfix 所架设的邮件转发代理，介绍如何使用 dovecot 实现 POP 和 IMAP 服务。最后简单介绍 Linux 系统下邮件客户端软件 Evolution 的使用。

第 11 章　数据库服务器 MySQL 的配置与架设

网络数据库主要为网络用户提供数据的存储、查询等服务，它以数据库管理系统为基础，广泛用于各类网站、搜索引擎、电子商务等方面。本章主要介绍数据库管理系统、SQL 语言等数据库方面的基础知识，然后介绍流行数据库服务器软件 MySQL 的安装、配置和使用，最后介绍如何以图形化的方式来方便地管理 MySQL 服务器。

通过本章的阅读，读者应掌握以下内容：

- ✧ 了解数据库管理系统、SQL 语言等基本概念
- ✧ MySQL 的安装
- ✧ MySQL 数据库的管理
- ✧ MySQL 的用户管理
- ✧ 熟悉 MySQL-Front 软件的使用

11.1　MySQL 概述

11.1.1　MySQL 简介

MySQL 是一个高性能的关系型数据库管理系统，具有功能强大、灵活性好的应用编程接口(API)和精巧的系统结构。MySQL 是现今世界上最受欢迎的开放源代码数据库，受到了广大软件用户的青睐。由于体积小、速度快、总体拥有成本低，尤其是开放源代码这一特点，Internet 上的许多中小型网站都选择 MySQL 作为网站数据库。

在 MySQL 数据库之前，mSQL 作为一种小型数据库得到广泛使用。mSQL 比较简单，进行简单 SQL 语句查询时速度较快，但是对于 SQL 语句的支持不够完善，基本不支持嵌套的 SQL 语句。此外，在安全性方面，mSQL 可靠性较差。

1996 年 5 月，瑞典的 TcX 公司开发出 MySQL 的最初版本，随后在 Internet 上公开发行。目前最新版本是 MySQL 5.1。这是一个快速、性价比高的数据库软件，并具有足够的伸缩性以应付任何数据库应用程序。MySQL 采用了全新的设计思路，引进了许多全新的概念，其主要特征如下。

- ✧ MySQL 是多线程、多用户的数据库系统，直接使用了系统核心的多线程内核，效率相当高，并意味着它可以采用多 CPU 体系结构。
- ✧ 在数据库客户端程序上，MySQL 为 C、C++、Eiffel 、Java、Perl、PHP、Python

和 TCL 等多种编程语言提供了各种不同的 API，极大方便了程序编写。

✧ MySQL 可运行在多种平台上，支持 AIX、FreeBSD、HP-UX、Linux、MacOS、Novell Netware、OpenBSD、OS/2 Wrap、Solaris、Windows 等多种操作系统，因此可以进行跨系统的开发。
✧ MySQL 性能高效稳定，它与当前的其他数据库相比性能都不差，Google、Cisco、Yahoo 等都用它作为数据库引擎。
✧ 有一个灵活安全的权限和口令系统，并且允许基于主机的认证。当与一个服务器连接时，所有的口令传送都被加密。
✧ 支持拥有上千万条记录的大型数据库处理，可以对某些包含 50 000 000 个记录的数据库使用 MySQL。
✧ 提供 TCP/IP、ODBC 和 JDBC 等多种数据库连接途径。
✧ 既可作为单独的应用程序使用在客户端服务器网络环境中，也可作为一个库嵌入到其他软件中提供多语言支持。

MySQL 由于设计上的革新，其整体性能与 mSQL 相比改进很多，尤其是复杂语句的执行速度有很大提高。

11.1.2 数据库管理系统简介

数据库管理系统(DBMS)是位于用户与操作系统之间的一层数据管理软件，用户对数据库数据的任何操作都是在 DBMS 管理下完成的，应用程序只有通过 DBMS 才能和数据库进行交互。数据库管理系统的主要功能包括以下几个方面。

✧ 数据库定义：DBMS 提供数据定义语言，用户通过它可以方便地对数据库中的数据对象进行定义。
✧ 数据操作：DBMS 提供数据操作语言，用户可以使用 DML 操纵数据实现对数据库的基本操作，如查询、插入、删除和修改等。
✧ 数据库建立和维护：数据库初始数据的输入、转换，数据库的转储、恢复，数据库的重组、性能监视及分析功能等。通常由实用程序完成这些功能。
✧ 数据库运行控制：数据库再建立、运用和维护时由数据库管理系统统一管理、统一控制，以保证数据的安全性、完整性、多用户对数据的并发使用及发生故障后的系统恢复。

11.2 SQL 语言发展简介

SQL(Structured Query Language)语言最早是由IBM的 San Jose Research Laboratory(圣约瑟研究实验室)为其关系数据库管理系统 SYSTEM R 开发的一种查询语言，它的前身是 SQUARE 语言。SQL 语言结构简洁，功能强大，简单易学，所以自从 1981 年 IBM 公司推出以来，SQL 语言得到了广泛应用，深受计算机工业界欢迎，被许多计算机及软件公司采用。经各公司不断修改、扩充和完善，SQL 语言最终发展成为关系数据库的标准语言。

目前，无论是Oracle、Sybase、Informix、SQL Server 等大型数据库管理系统，还是 Visual Foxporo、PowerBuilder 等微机上常用的数据库开发系统，都支持 SQL 语言作为查询语言。

1986 年 ANSI(美国国家标准局)数据库委员会批准 SQL 作为关系数据库语言的美国标准，同年又公布了 SQL 标准文本。1987 年 ISO(国际标准化组织)也通过了这一标准。此后 ANSI 又不断修改、完善该标准。SQL 成为国际标准语言后，各数据库厂家纷纷推出各自的 SQL 软件及其接口软件。于是，SQL 就成为大多数数据库共同的数据存取语言和标准接口，使不同数据库系统间的互操作有了共同基础，这个意义十分重大。

SQL 语言目前已成为数据库领域的一个主流语言。而 SQL 作为国际标准，对数据库以外的领域也产生了巨大影响。

11.3　MySQL 服务器的安装与管理

11.3.1　MySQL 的安装

应用实例导航——为 S 学校安装 MySQL 服务器

※场景呈现

S 学校需要架设网络数据库，选用 MySQL 软件。需要首先安装 MySQL 数据库软件，为了安全起见，需要创建 mysql 用户用于启动 MySQL 数据库。

※技术要领

(1) MySQL 数据库的安装。
(2) MySQL 数据库的启动。

MySQL 是一个使用广泛的数据库管理系统，Fedora 6 可以选择默认安装 MySQL 数据库，尽管如此，本节仍需介绍压缩包形式的 MySQL 的安装方法，此安装方法同样适用于 RedHat 等其他 Linux 版本，如果 Linux 已经附带安装 MySQL 则可略过本节的阅读。安装步骤如下。

❶ 将 MySQL 安装包 mysql-5.0.18.tar.gz 复制到/usr/local 后，使用下面命令解压缩该安装包，如图 11-1 所示，解压成功后生成/usr/local/openldap-2.3.38 目录。

```
tar -zxvf mysql-5.0.18.tar.gz
```

```
[root@cgsp local]# tar -zxvf mysql-5.0.18.tar.gz
mysql-5.0.18/
mysql-5.0.18/bdb/
mysql-5.0.18/bdb/Makefile.in
```

图 11-1　解压 MySQL 安装包

❷ 进入 MySQL 的安装目录，输入下面命令进行安装配置，如图 11-2 所示。

```
./configure --prefix=/usr/local/mysql
```

配置参数的说明如下:

--prefix=/usr/local/mysql 指定了 MySQL 的安装目录为/usr/local/mysql。

```
[root@cgsp mysql-5.0.18]# ./configure --prefix=/usr/local/mysql
checking build system type... x86_64-unknown-linux-gnu
checking host system type... x86_64-unknown-linux-gnu
checking target system type... x86_64-unknown-linux-gnu
checking for a BSD-compatible install... /usr/bin/install -c
```

图 11-2　配置 MySQL 安装包

图 11-3 显示配置 MySQL 安装包结束。

```
config.status: creating usr/Makefile
config.status: creating ib_config.h
config.status: executing depfiles commands

MySQL has a Web site at http://www.mysql.com/ which carries details on the
latest release, upcoming features, and other information to make your
work or play with MySQL more productive. There you can also find
information about mailing lists for MySQL discussion.

Remember to check the platform specific part of the reference manual for
hints about installing MySQL on your platform. Also have a look at the
files in the Docs directory.

Thank you for choosing MySQL!
```

图 11-3　配置 MySQL 安装包结束

❸ 配置完毕后，接着就是编译 MySQL 数据库安装包，输入下面命令编译，如图 11-4 所示。编译过程需要较长时间，一般的 PC 需要 6～10 分钟，服务器视性能而异。

```
Make
```

❹ 编译无误后，就可以安装 MySQL 数据库安装包了，输入下面命令进行安装，如图 11-5 所示。安装过程大概 1～2 分钟。

```
make install
```

```
[root@cgsp mysql-5.0.18]# make
cd libmysql; make link_sources
make[1]: Entering directory `/usr/local/mysql-5.0.18/libmysql'
set -x; \
  ss=`echo strmov.lo strxmov.lo strxnmov.lo strnmov.lo strmake.lo stre
int2str.lo str2int.lo strinstr.lo strcont.lo strcend.lo bcmp.lo ctype-
2str.lo strtoull.lo strtoll.lo llstr.lo my_vsnprintf.lo ctype.lo ctype
```

图 11-4　编译 MySQL 安装包

```
[root@cgsp mysql-5.0.18]# make install
make  install-recursive
make[1]: Entering directory `/usr/local/mysql-5.0.18'
Making install in .
make[2]: Entering directory `/usr/local/mysql-5.0.18'
make[3]: Entering directory `/usr/local/mysql-5.0.18'
make[3]: Nothing to be done for `install-exec-am'.
make[3]: Nothing to be done for `install-data-am'.
make[3]: Leaving directory `/usr/local/mysql-5.0.18'
make[2]: Leaving directory `/usr/local/mysql-5.0.18'
```

图 11-5　安装 MySQL 安装包

5 安装完毕后，以管理员 root 身份进入安装目录/usr/local/mysql 下，使用下面命令安装 MySQL 默认的数据表，如图 11-6 所示。这些表的作用大都用来管理用户权限，详细用途将在 11.3.3 节中介绍。

```
bin/mysql_install_db --user=mysql
```

```
[root@cgsp mysql-5.0.18]# cd /usr/local/mysql
[root@cgsp mysql]# bin/mysql_install_db --user=mysql
Installing all prepared tables
Fill help tables

To start mysqld at boot time you have to copy support-files/mysql.server
to the right place for your system

PLEASE REMEMBER TO SET A PASSWORD FOR THE MySQL root USER !
To do so, start the server, then issue the following commands:
/usr/local/mysql/bin/mysqladmin -u root password 'new-password'
/usr/local/mysql/bin/mysqladmin -u root -h cgsp password 'new-password'
See the manual for more instructions.

You can start the MySQL daemon with:
cd /usr/local/mysql ; /usr/local/mysql/bin/mysqld_safe &

You can test the MySQL daemon with the benchmarks in the 'sql-bench' direc
cd sql-bench ; perl run-all-tests

Please report any problems with the /usr/local/mysql/bin/mysqlbug script!

The latest information about MySQL is available on the web at
http://www.mysql.com
Support MySQL by buying support/licenses at https://order.mysql.com
```

图 11-6　安装 MySQL 数据库默认表

❻ 接着，因为用 mysql 用户启动 MySQL 数据库时要在某些目录下创建一些文件，因此将/usr/local/mysql/.目录和/usr/local/mysql/var 目录的用户赋给 mysql 用户，并将/usr/local/mysql/.目录的群组权限也赋给 mysql 组。然后使用下面命令启动 MySQL 数据库，如图 11-7 所示。

```
bin/mysqld_safe- -user=mysql &
```

“&”符号表示 mysqld 进程在后台运行，如果启动命令后不加上此符号，shell 命令行将挂起(suspend)，即该 shell 命令行不再接受任何其他命令。

```
[root@cgsp mysql]# chown -R root  .
[root@cgsp mysql]# chown -R mysql var
[root@cgsp mysql]# chgrp -R mysql .
[root@cgsp mysql]# bin/mysqld_safe --user=mysql &
```

图 11-7 启动 mysql 服务

❼ 如果需要开机即启动 mysqld 进程，可以将启动命令写入/etc/rc.local 文件，如图 11-8 所示。

```
#!/bin/sh
#
# This script will be executed *after* all the other init scripts.
# You can put your own initialization stuff in here if you don't
# want to do the full Sys V style init stuff.

touch /var/lock/subsys/local

/usr/local/mysql/bin/mysqld_safe --user=mysql
```

图 11-8 设置开机即启动 MySQL

点评与拓展： 为何用 mysql 启动 MySQL 数据库，而不用 root 用户呢？这个原因在前面章节多次谈过，比如 vsftpd、postfix 等，都是为了将进程的权限等级降低，使它不能操作其他目录或控制其他进程。

11.3.2 MySQL 数据库的管理

应用实例导航——S 学校的 MySQL 数据库管理

※场景呈现

S 学校安装好 MySQL 软件后，需要建立一个用于存放学生信息的数据库，数据库名字为 person，并在其中建立一张名字为 basicinfo 的表记录学生的基本信息，包括学号、名字、性别、住址、电话等信息。

另外，还需要为 MySQL 数据库的管理员设置密码，并在 basicinfo 表的名字字段上建立索引。

※技术要领

(1) MySQL 数据库管理员密码的管理。
(2) MySQL 数据库的创建和删除。
(3) MySQL 数据库中表的创建、复制、删除和修改。
(4) 表中数据的插入、删除和修改。
(5) MySQL 数据库索引的创建和删除。

1. MySQL 管理员密码修改和登录

如果是第一次安装 MySQL，访问数据库服务器的用户只能是管理员(即 root 用户)。默认情况下 root 的初始密码为空，本地连接 MySQL 时输入 mysql 即可。如果要修改 MySQL 管理员密码，需先退出 MySQL 命令状态，然后再使用下列命令。

```
mysqladmin -u root password seugrid
```

用上面的命令就将用户密码修改为 seugrid，如图 11-9 所示。

```
[root@sec local]# cd mysql
[root@sec mysql]# cd bin
[root@sec bin]# ls
comp_err            mysqlbinlog              mysql_explain_log          mysqltest
innochecksum        mysqlbug                 mysql_find_rows            mysqltestmanagerc
msql2mysql          mysqlcheck               mysql_fix_extensions       mysqltestmanager-pwgen
myisamchk           mysql_client_test        mysql_fix_privilege_tables mysql_tzinfo_to_sql
myisam_ftdump       mysql_config             mysqlhotcopy               mysql_waitpid
myisamlog           mysql_convert_table_format mysqlimport              mysql_zap
myisampack          mysql_create_system_tables mysql_install_db         perror
my_print_defaults   mysqld_multi             mysql_secure_installation  replace
mysql               mysqld_safe              mysql_setpermission        resolveip
mysqlaccess         mysqldump                mysqlshow                  resolve_stack_dump
mysqladmin          mysqldumpslow            mysql_tableinfo
[root@sec bin]# ./mysqladmin -u root password seugrid
```

图 11-9　修改 root 密码

若要以管理员 root 的身份连接到本机 MySQL，可以使用下列命令。

```
mysql -u root -p
```

在系统提示“Enter password:”后面输入密码即可，如图 11-10 所示。密码如果不正确则出现如图 11-11 所示的提示。

```
[root@sec bin]# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 233594 to server version: 5.0.18-log

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

图 11-10　登录 MySQL 时需要输入密码

```
[root@sec local]# mysql -u root -p
Enter password:
ERROR 1045 (28000): Access denied for user 'root'@'localhost' (using password: YES)
```

图 11-11　root 用户密码输入错误

2．数据库的创建和删除

MySQL 是个符合标准的关系型数据库管理系统，可以很好识别 SQL 语言。所以，和在许多其他关系型数据库管理系统中一样，SQL 的数据定义语言(DDL)也可以用来创建、删除 MySQL 中的数据库。用户登录 MySQL 后，所有 SQL 语句都是在命令提示符“mysql>”后输入，每个语句都以符号“；”或“\g”结束。另外，大小写字母皆可输入。

若要创建一个名为 person 的数据库，可以输入下列命令。

```
mysql>create database person;
```

该命令执行结果如图 11-12 所示。需要注意的是，默认情况下创建的数据库是保存在/var/lib/mysql 目录下，且系统不允许数据库重名。创建数据库后，可以使用下列命令查看当前所有数据库。

```
mysql>show databases;
```

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| gridsphere         |
| mysql              |
| seugrid            |
| test               |
+--------------------+
5 rows in set (0.00 sec)

mysql> create database person;
Query OK, 1 row affected (0.01 sec)
```

图 11-12　创建名为 person 的数据库

命令执行结果如图 11-13 所示。

为了演示删除数据库的操作，创建一个临时数据库 persontemp，如图 11-14 所示。然后可以通过下面命令删除数据库(包括删除该数据库中的所有表及表中数据)。使用删除命令后，再次使用 show databases 命令查看数据库，可以看到 persontemp 数据库已经被删除，如图 11-15 所示。

```
mysql> drop database persontemp;
```

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| gridsphere         |
| mysql              |
| person             |
| seugrid            |
| test               |
+--------------------+
6 rows in set (0.00 sec)

mysql>
```

图 11-13　查看当前所有可用数据库

```
mysql> create database persontemp;
Query OK, 1 row affected (0.01 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| gridsphere         |
| mysql              |
| person             |
| persontemp         |
| seugrid            |
| test               |
+--------------------+
7 rows in set (0.00 sec)
```

图 11-14　创建临时数据库

```
mysql> drop database persontemp;
Query OK, 0 rows affected (0.00 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| gridsphere         |
| mysql              |
| person             |
| seugrid            |
| test               |
+--------------------+
```

图 11-15　删除临时数据库

3．表的创建、复制、删除和修改

关系型数据库中的表用来存储数据，每个表由行和列组成，每一行为一个记录，每个记录可包含多个列。MySQL 可以使用 SQL 的数据定义语言(DDL)来创建、复制、删除和修改表结构。

数据库管理系统中包含多个数据库，如果要对某个数据库的表进行操作，首先要选择该数据库。若要选择前面所创建的 person 数据库，可使用下列命令。

```
mysql>use person;
```

命令执行结果如图 11-16 所示，shell 显示 Database changed，表示选择数据库成功。

```
mysql> use person;
Database changed
mysql> 
```

图 11-16　选择 person 数据库

在数据库中创建一个名为 basicinfo 的表，可输入如图 11-17 所示的命令。为便于输入和检查，可分行输入命令，每行用回车键结束，最后以“；”结束全部命令。

```
mysql> create table basicinfo (
    -> no varchar(7) not null,
    -> name varchar(20) not null,
    -> sex char(1) default 'm',
    ->  birthday date,
    -> phonenum varchar(15),
    ->  address varchar(30)
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql> 
```

图 11-17　创建 basicinfo 表

创建 basicinfo 表后，可用 describe 命令查看所建表的结构。具体命令格式如下：

```
mysql>describe basicinfo;
```

命令执行结果如图 11-18 所示。

为了方便用户创建新表，MySQL 还提供了复制表结构功能。可以使用下列 SQL 命令来复制表结构：

```
create  table 新表名称 like 源表名称;
```

例如若要将表 basicinfo 复制成另一个表 obasic，可以使用下列命令：

```
mysql>create table obasic like basicinfo;
```

```
mysql> describe basicinfo;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| no        | varchar(7)  | NO   |     |         |       |
| name      | varchar(20) | NO   |     |         |       |
| sex       | char(1)     | YES  |     | m       |       |
| birthday  | date        | YES  |     | NULL    |       |
| phonenum  | varchar(15) | YES  |     | NULL    |       |
| address   | varchar(30) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
6 rows in set (0.00 sec)

mysql>
```

图 11-18 查看 basicinfo 表

该命令执行结果如图 11-19 所示。

```
mysql> create table obasic like basicinfo;
Query OK, 0 rows affected (0.00 sec)

mysql> describe obasic;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| no        | varchar(7)  | NO   |     |         |       |
| name      | varchar(20) | NO   |     |         |       |
| sex       | char(1)     | YES  |     | m       |       |
| birthday  | date        | YES  |     | NULL    |       |
| phonenum  | varchar(15) | YES  |     | NULL    |       |
| address   | varchar(30) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
6 rows in set (0.00 sec)

mysql>
```

图 11-19 复制表

MySQL 还提供了删除数据库中一个或多个表的命令，命令格式如下：

```
drop  table 表名称 1[,表名称 2,…];
```

若要删除表 obasic，则可以使用下列命令：

```
mysql>drop table obasic;
```

该命令执行结果如图 11-20 所示。

若要修改表结构，比如增加、删除或修改表中字段，更改表的名称、类型，创建、撤销索引，等等，则可以使用 alter 命令，命令格式如下：

```
alter table 表名称 更改动作 1[,更改动作 2,…];
```

例如若要将表 basicinfo 中 address 字段的类型从 varchar(30)变为 varchar(35)，则可以使用下列命令：

```
mysql>alter table basicinfo modify address varchar(35);
```

```
mysql> drop table obasic;
Query OK, 0 rows affected (0.00 sec)

mysql> show tables;
+--------------------+
| Tables_in_person |
+--------------------+
| basicinfo          |
+--------------------+
1 row in set (0.00 sec)

mysql>
```

图 11-20　删除表

若要在表 basicinfo 中增加一个字段 email，则可以使用下列命令：

```
mysql>alter table basicinfo add email varchar(20);
```

上述两个命令的执行结果如图 11-21 所示。

```
mysql> alter table basicinfo modify address varchar(35);
Query OK, 0 rows affected (0.04 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> alter table basicinfo add email varchar(20);
Query OK, 0 rows affected (0.03 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> describe basicinfo;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| no       | varchar(7)  | NO   |     |         |       |
| name     | varchar(20) | NO   |     |         |       |
| sex      | char(1)     | YES  |     | m       |       |
| birthday | date        | YES  |     | NULL    |       |
| phonenum | varchar(15) | YES  |     | NULL    |       |
| address  | varchar(35) | YES  |     | NULL    |       |
| email    | varchar(20) | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
7 rows in set (0.00 sec)

mysql>
```

图 11-21　更改字段类型和增加字段

若要将表 basicinfo 中 phonenum 字段的名称变为 cellphone，且将该字段类型改为 varchar(15)，则可以使用下列命令：

```
mysql>alter table basicinfo change phonenum cellphone varchar(15);
```

若要在表 basicinfo 中删除一个字段 email，则可以使用下列命令：

```
mysql>alter table basicinfo drop email;
```

上述两个命令的执行结果如图 11-22 所示。

```
mysql> alter table basicinfo change phonenum cellphone varchar(15);
Query OK, 0 rows affected (0.04 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> alter table basicinfo drop email;
Query OK, 0 rows affected (0.03 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> describe basicinfo;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| no        | varchar(7)  | NO   |     |         |       |
| name      | varchar(20) | NO   |     |         |       |
| sex       | char(1)     | YES  |     | m       |       |
| birthday  | date        | YES  |     | NULL    |       |
| cellphone | varchar(15) | YES  |     | NULL    |       |
| address   | varchar(35) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
6 rows in set (0.00 sec)

mysql>
```

图 11-22　更改字段名及类型和删除字段

若要将表 basicinfo 的名称改为 infomation，则可以使用下列命令：

```
mysql>alter table basicinfo rename to infomation;
```

命令执行结果如图 11-23 所示。

```
mysql> alter table basicinfo rename to infomation;
Query OK, 0 rows affected (0.00 sec)

mysql> show tables;
+-------------------+
| Tables_in_person  |
+-------------------+
| infomation        |
+-------------------+
1 row in set (0.00 sec)

mysql>
```

图 11-23　更改表名称

4．表中数据的插入、删除和修改

建立数据库和表后，紧接着就是在表中存储数据。在 MySQL 中，通常使用数据操作语言(DML)来插入、删除和修改表中记录。本节介绍对数据库表中数据进行操作的各种命令，包括数据的插入、删除和修改。

首先介绍表中数据的插入命令，命令格式如下：

```
insert into 表名称(字段名 1，字段名 2，…)
values (字段 1 的值，字段 2 的值，…);
```

例如若要在表 infomation 中插入一组数据，可以使用下列命令：

```
mysql> insert into infomation (no,name,sex,birthday,cellphone,address)
    -> values ('0081','YY','f','1982-01-08','025-6754313','NNU');
```

命令执行结果如图 11-24 所示。由图可知，插入记录后可用 select 语句查看所插记录是否正确。

```
mysql> insert into infomation (no,name,sex,birthday,cellphone,address)
    -> values ('0081','YY','f','1982-01-08','025-6754313','NNU');
Query OK, 1 row affected (0.01 sec)

mysql> select * from infomation;
+------+------+------+------------+-------------+---------+
| no   | name | sex  | birthday   | cellphone   | address |
+------+------+------+------------+-------------+---------+
| 0081 | YY   | f    | 1982-01-08 | 025-6754313 | NNU     |
+------+------+------+------------+-------------+---------+
1 row in set (0.00 sec)

mysql>
```

图 11-24　插入记录

insert 命令还可以连续插入多条记录，比如要在表 infomation 中插入两条记录，可以使用下面 insert 的缩写命令格式：

```
mysql> insert into infomation values
-> ('0071','BOB','m','1986-09-11','025-8374313','SEU'),
-> ('0099','TT','f','1985-06-09','025-6434313','NJUPT');
```

命令执行结果如图 11-25 所示。

```
mysql> insert into infomation values
    -> ('0071','BOB','m','1986-09-11','025-8374313','SEU'),
    -> ('0099','TT','f','1985-06-09','025-6434313','NJUPT');
Query OK, 2 rows affected (0.00 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> select * from infomation;
+------+------+------+------------+-------------+---------+
| no   | name | sex  | birthday   | cellphone   | address |
+------+------+------+------------+-------------+---------+
| 0081 | YY   | f    | 1982-01-08 | 025-6754313 | NNU     |
| 0071 | BOB  | m    | 1986-09-11 | 025-8374313 | SEU     |
| 0099 | TT   | f    | 1985-06-09 | 025-6434313 | NJUPT   |
+------+------+------+------------+-------------+---------+
3 rows in set (0.00 sec)

mysql>
```

图 11-25　连续插入记录

其次介绍表中数据的修改命令，它用于表中数据的更新，命令格式如下：

```
UPDATE 表名称 SET 字段名 1=字段值 1[,字段名 2=字段值 2,…]
WHERE 条件表达式;
```

若要修改表 infomation 中 no 字段值为 0071 的记录，将其对应的 birthday 值改为 19870505，cellphone 值改为 025-8379324，可以使用下列命令：

```
mysql> update infomation set birthday=19870505 , cellphone='025－8379324'
where no='0071';
```

命令执行结果如图 11-26 所示。

```
mysql> update infomation set birthday=19870505, cellphone='025-8379324' where no='0071';
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from infomation where no='0071';
+------+------+------+------------+-------------+---------+
| no   | name | sex  | birthday   | cellphone   | address |
+------+------+------+------------+-------------+---------+
| 0071 | BOB  | m    | 1987-05-05 | 025-8379324 | SEU     |
+------+------+------+------------+-------------+---------+
1 row in set (0.00 sec)

mysql>
```

图 11-26　修改记录

最后介绍表中数据的删除命令，命令格式如下：

```
DELETE FROM 表名称 WHERE 条件表达式;
```

若要删除表 infomation 中 no 字段值为 0099 的记录，可以使用下列命令：

```
mysql> delete from infomation where no＝'0099'
```

命令执行结果如图 11-27 所示。

```
mysql> delete from infomation where no='0099';
Query OK, 1 row affected (0.01 sec)

mysql> select * from infomation;
+------+------+------+------------+-------------+---------+
| no   | name | sex  | birthday   | cellphone   | address |
+------+------+------+------------+-------------+---------+
| 0081 | YY   | f    | 1982-01-08 | 025-6754313 | NNU     |
| 0071 | BOB  | m    | 1987-05-05 | 025-8379324 | SEU     |
+------+------+------+------------+-------------+---------+
2 rows in set (0.00 sec)

mysql>
```

图 11-27　删除记录

5．索引的创建和删除

何谓数据库索引呢？它类似于书籍中的索引。在书籍中，索引使读者不必翻阅整本书就能迅速地找到所要查阅的条目。在数据库中，索引也允许数据库程序迅速地找到表中的数据，而不必扫描整个数据库，这是一种加快检索表中数据的方法。

MySQL 提供了创建和删除索引的命令，使用 CREATE INDEX 命令语句可以向已经存

在的表中添加索引，其基本命令格式如下：

```
CREATE [UNIQUE] INDEX 索引名 ON 表名称 (字段名 1[(长度)],…);
```

若要为表 information 的 name 字段创建名为 iname 的索引，可以使用下列命令格式：

```
mysql> create index iname on information (name);
```

命令执行结果如图 11-28 所示。

```
mysql> create index iname on information (name);
Query OK, 2 rows affected (0.03 sec)
Records: 2  Duplicates: 0  Warnings: 0
```

图 11-28　创建索引

DROP INDEX 命令语句用于删除索引，其基本命令格式如下：

```
DROP INDEX 索引名 ON 表名称;
```

若要删除表 information 中名为 iname 的索引，可以使用下列命令格式：

```
mysql>drop index iname on information;
```

命令执行结果如图 11-29 所示。

```
mysql> drop index iname on information;
Query OK, 2 rows affected (0.08 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql>
```

图 11-29　删除索引

点评与拓展：每次修改完数据库中的表后，都最好用 describe 命令来检查一下修改结果。另外，使用 UPDATE 语句修改表中数据时，如果忘了用 WHERE 子句进行限制，很有可能导致大量数据被破坏。

11.3.3　MySQL 服务器的用户管理

应用实例导航——S 学校 MySQL 用户管理

※场景呈现

为了远程对 S 学校的 person 数据库进行管理，需要建立一个名字为 guest 的用户，允许该用户从任意主机都能登录，并对它赋于一定权限，使其可以对 person 数据库进行一切操作，具备与管理员对 person 数据库一样的权限。

※技术要领

(1) MySQL 数据库访问控制原理。
(2) MySQL 数据库用户的创建和删除。
(3) MySQL 数据库用户的授权和撤销。

1．MySQL 服务器的访问控制

首次安装 MySQL 时，安装程序会在数据库 mysql 中设置 5 个授权表，即 user、db、host、tables_ priv、columnts_ priv。这 5 个表共同决定可以连接到数据库服务器的用户、从哪连接以及其可以执行的操作。在初始化后，有 3 个表(host、ables_priv、columnts_ priv)为空，则 user、db 这两个表就决定了 MySQL 的默认访问规则。

2．用户的创建和删除

user 表从用户角度规定了用户账户的权限，若要查看 MySQL 数据库中表 user 前四个字段的内容，可以使用下列命令格式：

```
mysql>select host,user,password,select_priv from mysql.user;
```

命令执行结果如图 11-30 所示，由图可知，root 用户具有从本地主机和主机 seugrid 登录的权限，且密码以加密形式存放。

```
mysql> select host,user,password,select_priv from mysql.user;
+-----------+---------+-------------------------------------------+-------------+
| host      | user    | password                                  | select_priv |
+-----------+---------+-------------------------------------------+-------------+
| localhost | root    | *A05ACE78A574B7D63E18D4F155F3B09BC1470797 | Y           |
| seugrid   | root    |                                           | Y           |
| seugrid   |         |                                           | N           |
| localhost |         |                                           | N           |
|           | seugrid | *A05ACE78A574B7D63E18D4F155F3B09BC1470797 | N           |
+-----------+---------+-------------------------------------------+-------------+
5 rows in set (0.00 sec)

mysql>
```

图 11-30　mysql.user 表的内容

db 表规定了数据库的被访问权限，若要查看 MySQL 数据库中表 db 前四个字段的内容，可以使用下列命令格式：

```
mysql>select host,db,user,select_priv from mysql.db;
```

命令执行结果如图 11-31 所示，由图可知，seugrid 数据库允许从任意主机的用户访问。

```
mysql> select host,db,user,select_priv from mysql.db;
+------+----------+------+-------------+
| host | db       | user | select_priv |
+------+----------+------+-------------+
| %    | test     |      | Y           |
| %    | test\_%  |      | Y           |
| %    | seugrid  |      | Y           |
+------+----------+------+-------------+
3 rows in set (0.01 sec)

mysql>
```

图 11-31　mysql.db 表前 4 个字段内容

那么如何为数据库开放一个用户，使其可以从任意主机连接到数据库服务器呢？若要创建一个新用户 guest 满足上述要求，可以使用下列命令：

```
mysql>insert into mysql.user (host,user,password)
        ->values ('%','guest',password('guest'));
```

这里需要用到 password( )函数为密码加密，这样表 user 的 password 字段中保存的就是加密过的密码。命令执行结果如图 11-32 所示，由图可知，user 表中已经成功添加了 guest 用户。

```
mysql> insert into mysql.user (host,user,password)
    -> values ('%','guest',password('guest'));
Query OK, 1 row affected, 3 warnings (0.00 sec)

mysql> select host,user,password,select_priv from mysql.user;
+-----------+---------+-------------------------------------------+-------------+
| host      | user    | password                                  | select_priv |
+-----------+---------+-------------------------------------------+-------------+
| localhost | root    | *A05ACE78A574B7D63E18D4F155F3B09BC1470797 | Y           |
| seugrid   | root    |                                           | Y           |
| seugrid   |         |                                           | N           |
| localhost |         |                                           | N           |
|           | seugrid | *A05ACE78A574B7D63E18D4F155F3B09BC1470797 | N           |
| %         | guest   | *11DB58B0DD02E290377535868405F11E4CBEFF58 | N           |
+-----------+---------+-------------------------------------------+-------------+
6 rows in set (0.00 sec)
```

图 11-32　创建新用户

接下来需要重载 MySQL 授权表，可以使用下列命令实现：

```
mysql>flush privileges;
```

命令执行结果如图 11-33 所示。

guest 用户创建成功后，就可以从远程客户端使用下列命令连接数据库服务器，进而用来测试新建用户是否可用。

```
mysql -h 172.18.12.179 -u guest -p
```

```
mysql> flush privileges;
Query OK, 0 rows affected (0.01 sec)

mysql>
```

图 11-33　重载授权表

在用户正确输入密码后，就可成功连接到数据库服务器了。使用命令 show databases 可以查看用户当前可以使用的数据库。命令执行结果如图 11-34 所示。

```
[root@seugrid2 ~]# mysql -h 172.18.12.179 -u guest -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 233835 to server version: 5.0.18-log

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| seugrid            |
| test               |
+--------------------+
3 rows in set (0.00 sec)
```

图 11-34　远程客户端连接数据库服务器

如果要关闭远程主机连接数据库服务器的功能，只需删除用户 guest 即可，可以使用下列命令实现：

```
mysql>delete from mysql.user where user='guest';
```

删除后需要用命令 flush privileges 重载 MySQL 授权表，命令执行结果如图 11-35 所示。

```
mysql> delete from mysql.user where user='guest';
Query OK, 1 row affected (0.00 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.00 sec)

mysql> select host,user,password,select_priv from mysql.user;
+-----------+---------+-------------------------------------------+-------------+
| host      | user    | password                                  | select_priv |
+-----------+---------+-------------------------------------------+-------------+
| localhost | root    | *A05ACE78A574B7D63E18D4F155F3B09BC1470797 | Y           |
| seugrid   | root    |                                           | Y           |
| seugrid   |         |                                           | N           |
| localhost |         |                                           | N           |
|           | seugrid | *A05ACE78A574B7D63E18D4F155F3B09BC1470797 | N           |
+-----------+---------+-------------------------------------------+-------------+
5 rows in set (0.00 sec)

mysql>
```

图 11-35　删除 guest 用户

3. 用户权限的授予和撤销

为了方便地管理用户的权限，MySQL 数据库提供了授予和撤销用户权限的命令，下面举例说明这些命令的用法。

若要授予用户 guest 可从任意主机连接到数据库服务器，并可完全访问 person 数据库的权限，可以使用下列命令：

```
mysql>grant all on person.* to guest@'%' identified by 'guest';
```

命令执行结果如图 11-36 所示。

```
mysql> grant all on person.* to guest@'%' identified by 'guest';
Query OK, 0 rows affected (0.02 sec)

mysql> select host,db,user,select_priv from mysql.db;
+------+---------+-------+-------------+
| host | db      | user  | select_priv |
+------+---------+-------+-------------+
| %    | test    |       | Y           |
| %    | test\_% |       | Y           |
| %    | seugrid |       | Y           |
| %    | person  | guest | Y           |
+------+---------+-------+-------------+
4 rows in set (0.00 sec)
```

图 11-36　将 person 数据库授权给 guest 用户

若要新建一个用户 lab，使其可以从子网 172.18.12.0 中任何主机连接到数据库服务器，进而可读取 person 数据库内容，可使用下列命令格式：

```
mysql>grant select on person.* to lab@' 172.18.12.*' identified by 'grid';
```

命令执行结果如图 11-37 所示。这条命令表示，lab 用户以 grid 为密码可以从子网 172.18.12.0 中的任何主机连接到数据库服务器，并可访问 person 数据库。

```
mysql> grant select on person.* to lab@'172.18.12.*' identified by 'grid';
Query OK, 0 rows affected (0.00 sec)

mysql> select host,db,user,select_priv from mysql.db;
+-------------+---------+-------+-------------+
| host        | db      | user  | select_priv |
+-------------+---------+-------+-------------+
| %           | test    |       | Y           |
| %           | test\_% |       | Y           |
| %           | seugrid |       | Y           |
| %           | person  | guest | Y           |
| 172.18.12.* | person  | lab   | Y           |
+-------------+---------+-------+-------------+
5 rows in set (0.00 sec)
```

图 11-37　授予用户不同级别访问权限

可以使用下列命令查看表 mysql.user 中与用户 lab 有关的记录，以检验上一条命令：

```
mysql>select  host,user,password,select_priv  from  mysql.user  where
```

```
user='lab';
```

命令执行结果如图 11-38 所示。由图可见，MySQL 创建了用户 lab 并设置了密码，lab 可以从子网 172.18.12.0/24 中任一主机连接到数据库服务器，但授权表 user 中与 lab 对应的用以设置全局权限的各个字段值都为 N(如本例中的 select_ priv 字段)，所以 MySQL 并没有授予 lab 任何全局权限。

```
mysql> select host,user,password,select_priv from mysql.user where user='lab';
+-------------+------+-------------------------------------------+-------------+
| host        | user | password                                  | select_priv |
+-------------+------+-------------------------------------------+-------------+
| 172.18.12.* | lab  | *73B214E38F96B6940CA2B2038ED48CB58FCCC864 | N           |
+-------------+------+-------------------------------------------+-------------+
1 row in set (0.00 sec)
```

图 11-38　查看表 mysql.user 中的内容

查看表 mysql.db 中与用户 lab 相关的记录可以使用如下命令：

```
mysql>select host,db,user,select_priv from mysql.db;
```

命令执行结果如图 11-39 所示。由图可知，授权表 db 中与用户 lab 对应的用以设置权限的 select_priv 字段值为 Y，所以用户 lab 有权限浏览数据库 person 中的内容。

```
mysql> grant select on person.* to lab@'172.18.12.*' identified by 'grid';
Query OK, 0 rows affected (0.00 sec)

mysql> select host,db,user,select_priv from mysql.db;
+-------------+---------+-------+-------------+
| host        | db      | user  | select_priv |
+-------------+---------+-------+-------------+
| %           | test    |       | Y           |
| %           | test\_% |       | Y           |
| %           | seugrid |       | Y           |
| %           | person  | guest | Y           |
| 172.18.12.* | person  | lab   | Y           |
+-------------+---------+-------+-------------+
5 rows in set (0.00 sec)
```

图 11-39　查看 mysql.db 表中的内容

除了创建新用户以外，还可以对已有用户的权限进行变更。若要授予用户 guest 从本地连接到数据库服务器并对 person 数据库可完全访问的权限，且可以将其所有权限授予别的用户，则可使用下列命令：

```
mysql>grant all on person.* to guest@localhost identified by 'guest'
->with grant option;
```

命令执行结果如图 11-40 所示。

为用户 guest 授权后，可以使用下列命令检查授予该用户的权限：

```
mysql> show grants for guest@localhost;
```

命令执行结果如图 11-41 所示。

```
mysql> grant all on person.* to guest@localhost identified by 'guest'
    -> with grant option;
Query OK, 0 rows affected (0.00 sec)
```

图 11-40　授予 guest 用户管理权限

```
mysql> show grants for guest@localhost;
+--------------------------------------------------------------------------------------------------+
| Grants for guest@localhost                                                                       |
+--------------------------------------------------------------------------------------------------+
| GRANT USAGE ON *.* TO 'guest'@'localhost' IDENTIFIED BY PASSWORD '*11DB58B0DD02E290377535868405F11E4CBEFF58' |
| GRANT ALL PRIVILEGES ON `person`.* TO 'guest'@'localhost' WITH GRANT OPTION                      |
+--------------------------------------------------------------------------------------------------+
2 rows in set (0.00 sec)
```

图 11-41　检查 guest 用户管理权限

与上一命令相反，若要撤销用户 guest@localhost 对 person 数据库所有的创建、删除数据库表的权限，可以使用下列命令：

```
mysql>revoke create,drop on person.* from guest@localhost;
```

命令执行结果如图 11-42 所示。

```
mysql> revoke create,drop on person.* from guest@localhost;
Query OK, 0 rows affected (0.00 sec)
```

图 11-42　撤销 guest 的创建和删除表的权限

点评与拓展： MySQL 提供了两种修改授权表中访问权限的方法：一是用 INSERT、UPDATE、DELETE 等 DML 语句来修改信息；二是用 GRANT 和 REVOKE 语句。第一种方法虽然直观，但各授权表中字段数很多，容易出错，所以通常使用第二种方法。

11.4　使用 MySQL-Front 软件图形化管理 MySQL

应用实例导航——利用 MySQL-Front 管理 S 学校 MySQL 数据库

※场景呈现

为了方便地对 MySQL 数据库进行管理，S 学校利用 MySQL-Front 对 MySQL 数据库服务器以图形化的方式进行管理，使用 MySQL-Front 执行在 person 数据库中添加和删除表、添加和删除表的字段和数据等操作。

在熟悉 MySQL-Front 的基本操作的基础上，学校需要建一张学生选课表，名字为 course，里面记录了学生的选课情况，字段包括：学号、课程名 1、课程名 2、……等。course 表与

学生基本信息 information 表通过学生学号相联系；要求利用 MySQL-Front 软件实现两个表的连接操作，列出 course 表和 information 表中都有记录项的学生信息。

※技术要领

(1) MySQL-Front 的基本操作。
(2) 利用 MySQL-Front 实现数据库表之间的连接操作。

11.4.1　MySQL-Front 软件的基本操作

本节介绍 Windows 下常用的图形化管理 MySQL 数据库的软件：MySQL-Front 是一款实用的管理 MySQL 的应用程序，主要功能包括多文档界面的编写，提供拖拽方式管理数据库和表格，可提供对域和数据记录的编辑、增加、删除功能，提供执行的 SQL 脚本的功能，提供与外程序接口，以及保存数据到 CSV 文件等功能。

从本书光盘或者 Internet 下载，可以获得安装包 MySQL-Front 3.2 版本，双击 step by step 式进行安装，安装过程不再详述。

❶ MySQL-Front 安装完毕后，双击打开 MySQL-Front，弹出如图 11-43 所示的初始界面，首先需要输入登录会话的名称，该值不能省略。

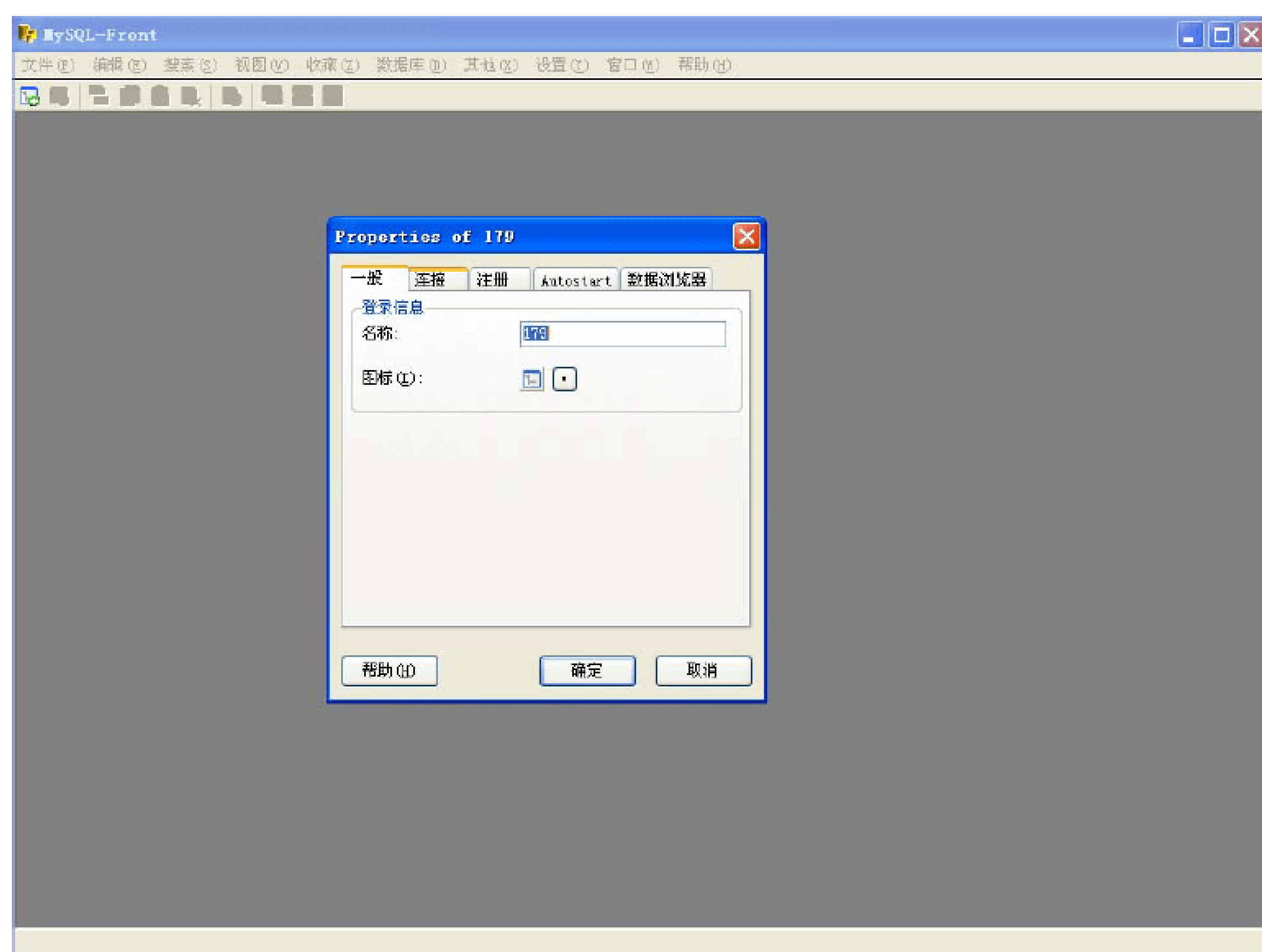


图 11-43　MySQL-Front 的初始界面

❷ 切换至【连接】选项卡，进行连接服务器 IP 地址和端口的设定，本例的 MySQL 数据库 IP 地址为 172.18.12.179，端口默认为 3306，如图 11-44 所示。其他选项都使用默认的方式即可。

❸ 在设定登录用户前，需要首先用命令行创建一个，将上节所创建的 person 数据库授权给 seugrid 用户，并使 seugrid 用户对 person 数据库具备管理员权限，如图 11-45 所示。然后切

换至【注册】选项卡，进行连接服务器的用户名和密码的设定，将刚才创建的 seugrid 用户填入其中，如图 11-46 所示。

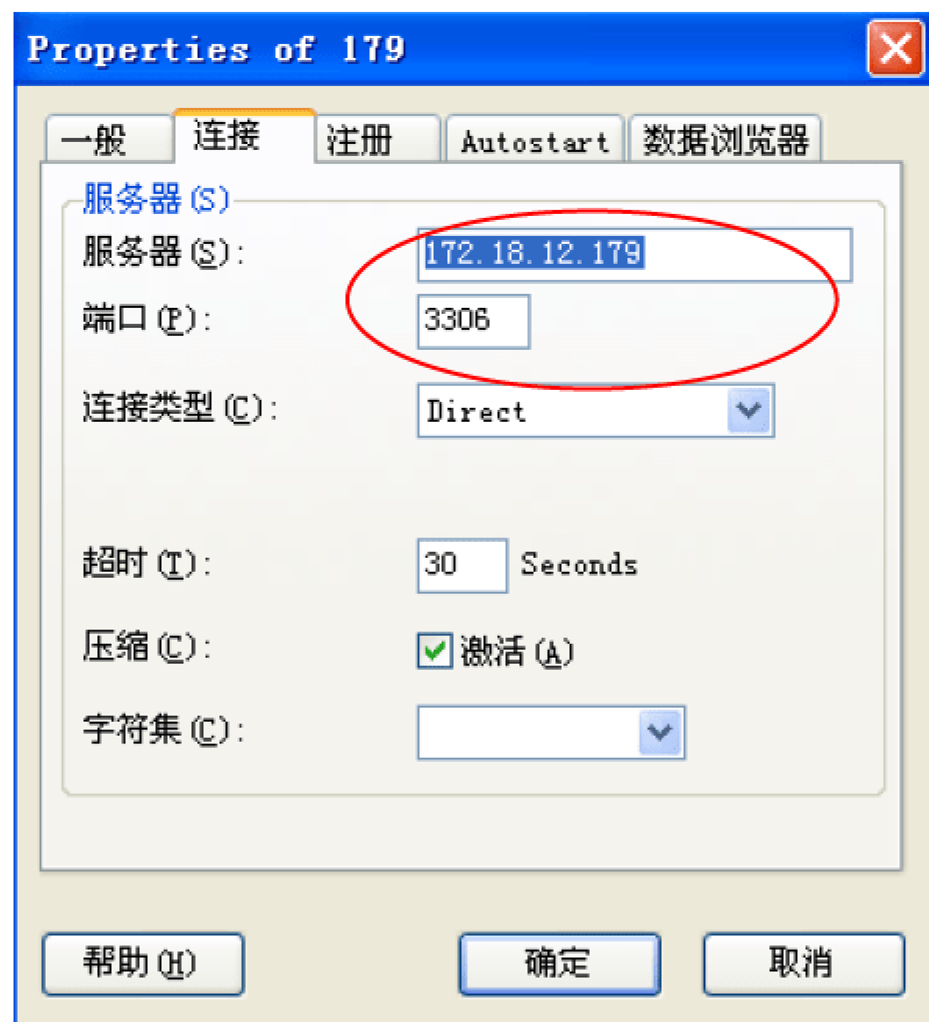


图 11-44　连接服务器和端口设定

```
mysql> grant all on person.* to seugrid@'%' identified by 'seugrid' with grant option;
Query OK, 0 rows affected (0.00 sec)

mysql>
```

图 11-45　将 seugrid 用户授权为外部主机访问用户

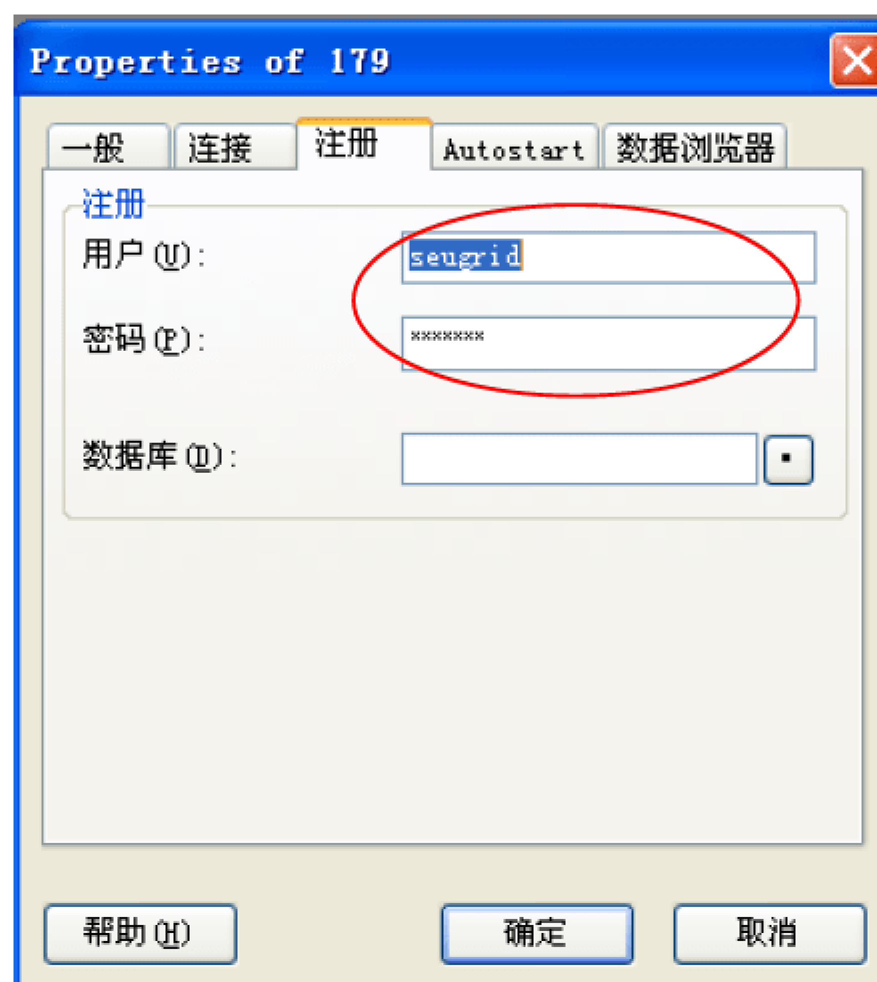


图 11-46　MySQL-Front 设定登录用户名

❹ 对上述的【连接】和【注册】选项卡设定完毕后，单击【确定】按钮，弹出如图 11-47 所示的对话框，单击【打开】按钮即可连接 MySQL 数据库服务器。

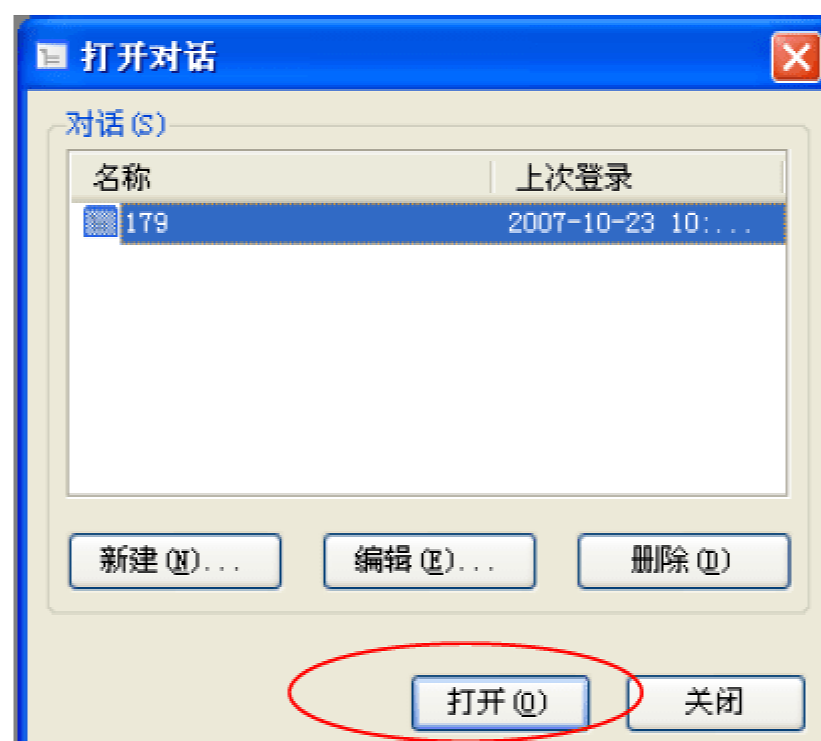


图 11-47　连接设定的数据库

❺ 连接成功后，弹出如图 11-48 所示窗口，左边区域列出了 172.18.12.179 这台服务器上所有的登录用户有权限访问的数据库，步骤(3)授权的 person 数据库也在其内；右边区域可以显示选定数据库表的各种信息，通过右边上方的按钮进行选择，如图 11-48 的标注部分。其中【对象浏览器】显示表的字段名称、类型、是否可以空以及默认值等属性，如图 11-49 所示；【数据浏览器】显示选定表内的所有数据，如图 11-50 所示；【SQL 编辑器】提供用户输入 SQL 语言的界面，用户可以在此处输入 SQL 对选定的表进行操作，效果在 shell 上输入 SQL 语句一样。

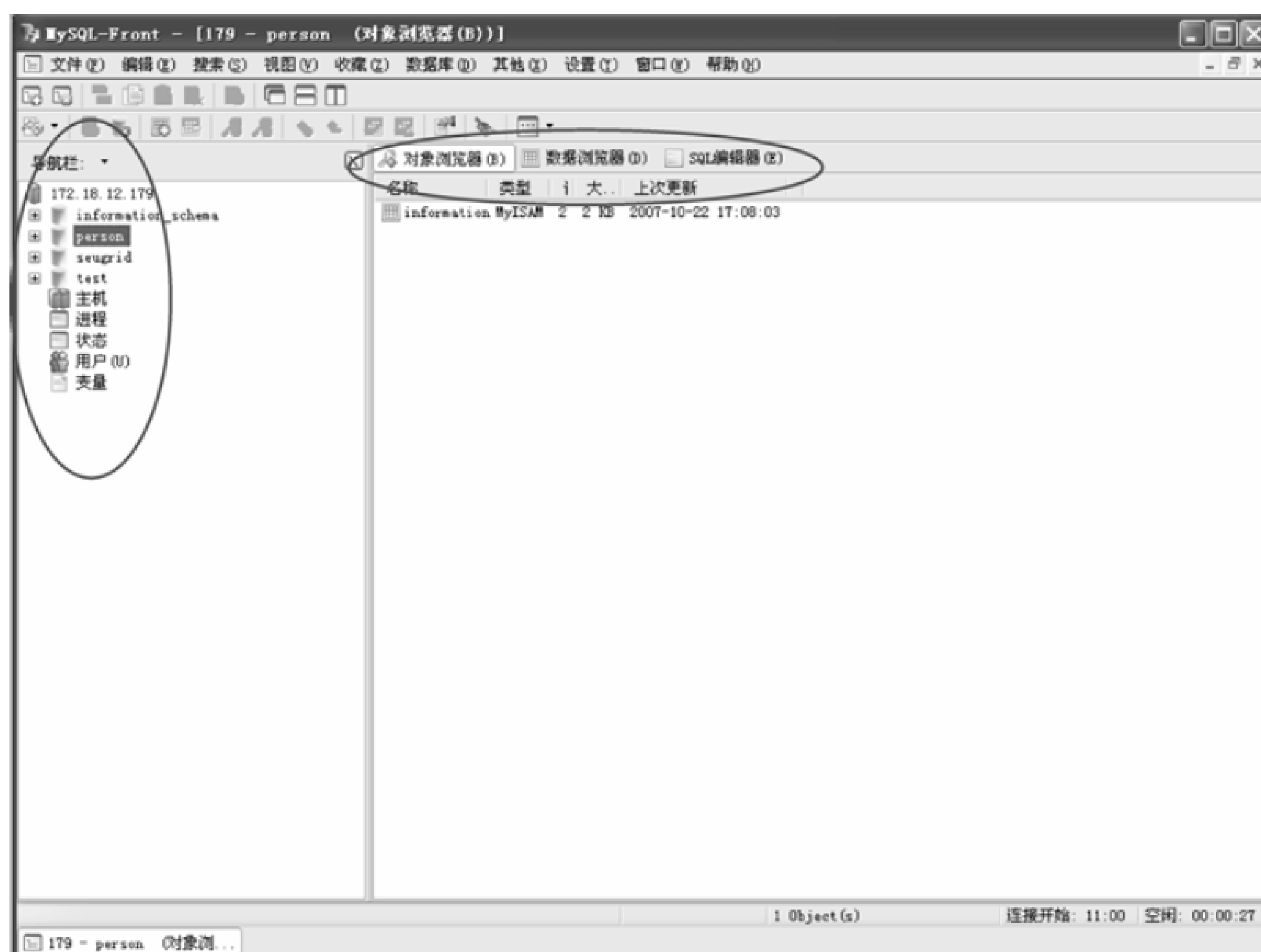


图 11-48　连接 MySQL 服务器成功

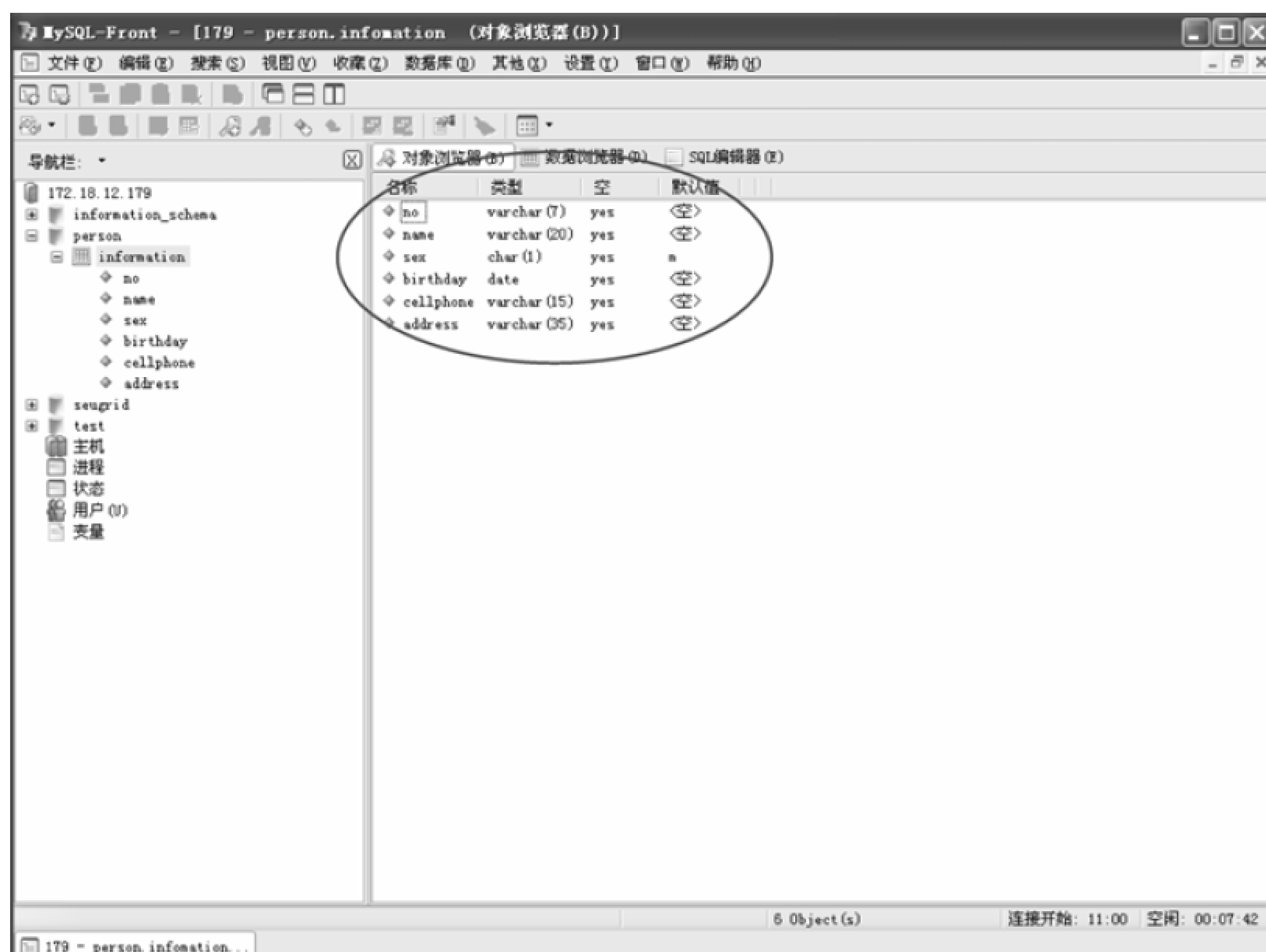


图 11-49　显示数据库表的字段信息

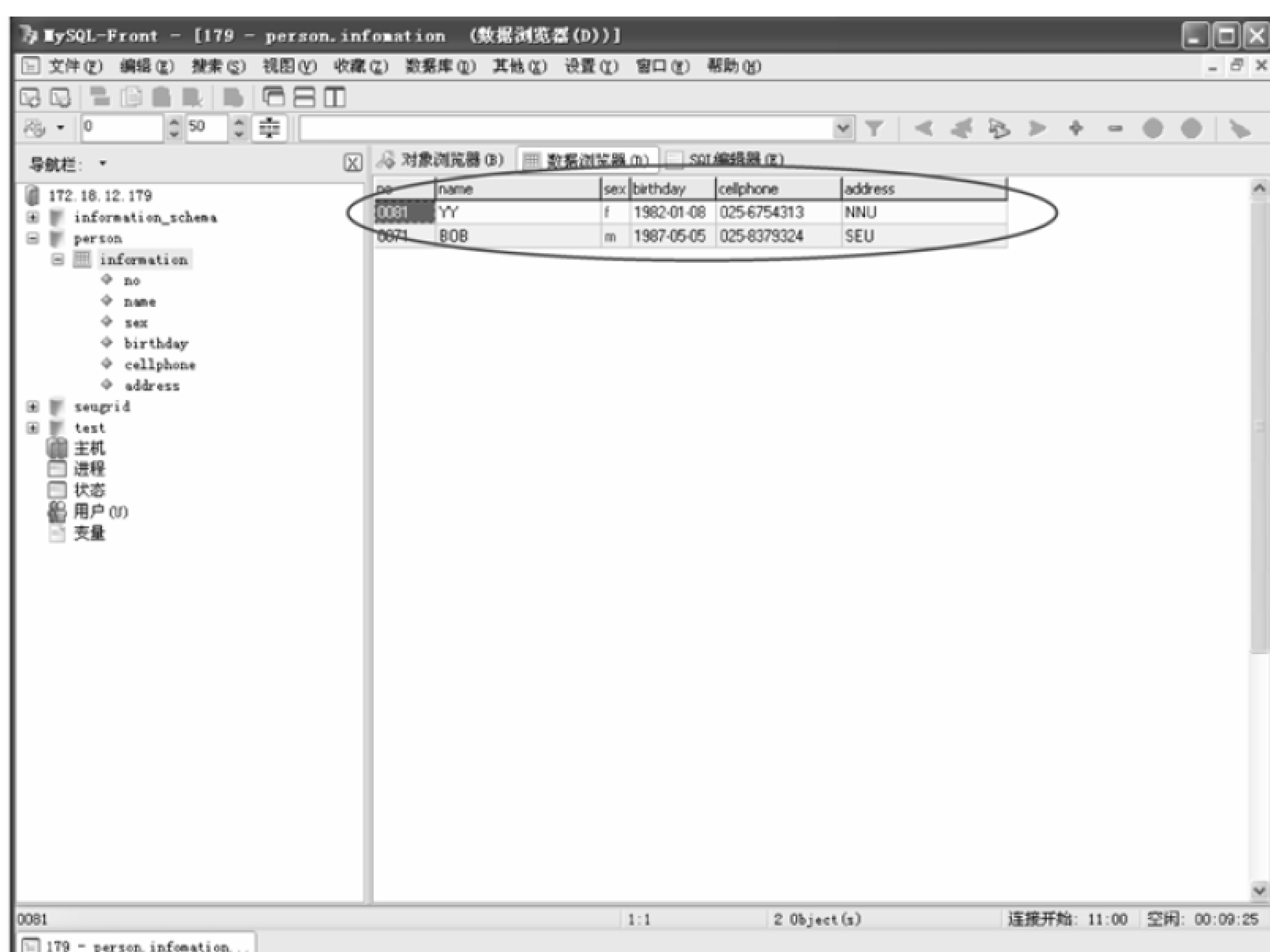


图 11-50　显示数据库表的数据信息

❻ MySQL-Front 提供了为数据库表新建字段和索引的功能，在图 11-48 的空白处右击，选择【新建】→【字段】命令就弹出如图 11-51 所示的对话框，在其中输入字段信息，【位置】

记录该字段位于现存的哪个字段之后，【名称】为字段名称，【类型】为字段对应的数据类型，【长度】为字段的最大长度，图示的选择相当于 SQL 语言的 varchar(15)。字段属性设定完毕后，单击【确定】按钮就可以看到在【对象浏览器】列表中出现了刚才添加的名称为 email 的字段，如图 11-52 标注部分所示。

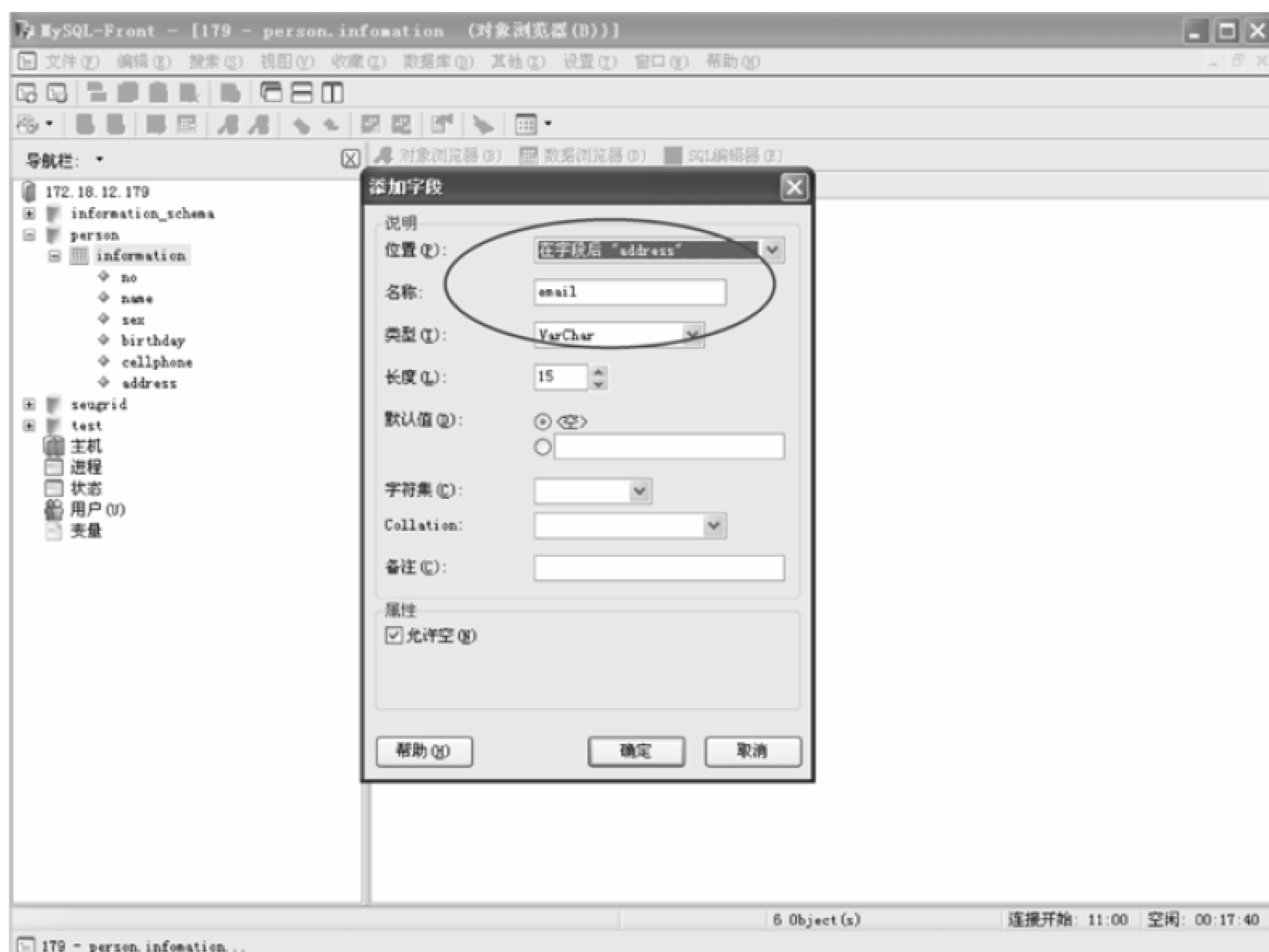


图 11-51　在表 information 中添加字段



图 11-52　email 字段添加成功

❼ MySQL-Front 还提供了为数据库表添加数据的功能，在图 11-49 所示的数据列表上右击选择【插入记录】命令，弹出如图 11-53 所示的编辑数据界面，可以图形化的输入表数据信息。

❽ MySQL-Front 还提供以各种形式输出数据库表内容，比如 HTML、access 数据库、CVS 等格式。比如，我们要以 HTML 格式输出 information 表的内容，只要选中 information 表，右击选择【输出】→【HTML 文件】命令，保存 information.html 文件，就得到如图 11-54 所示的以 HTML 格式显示数据库表的 HTML 文件。



图 11-53　在表 information 中添加数据

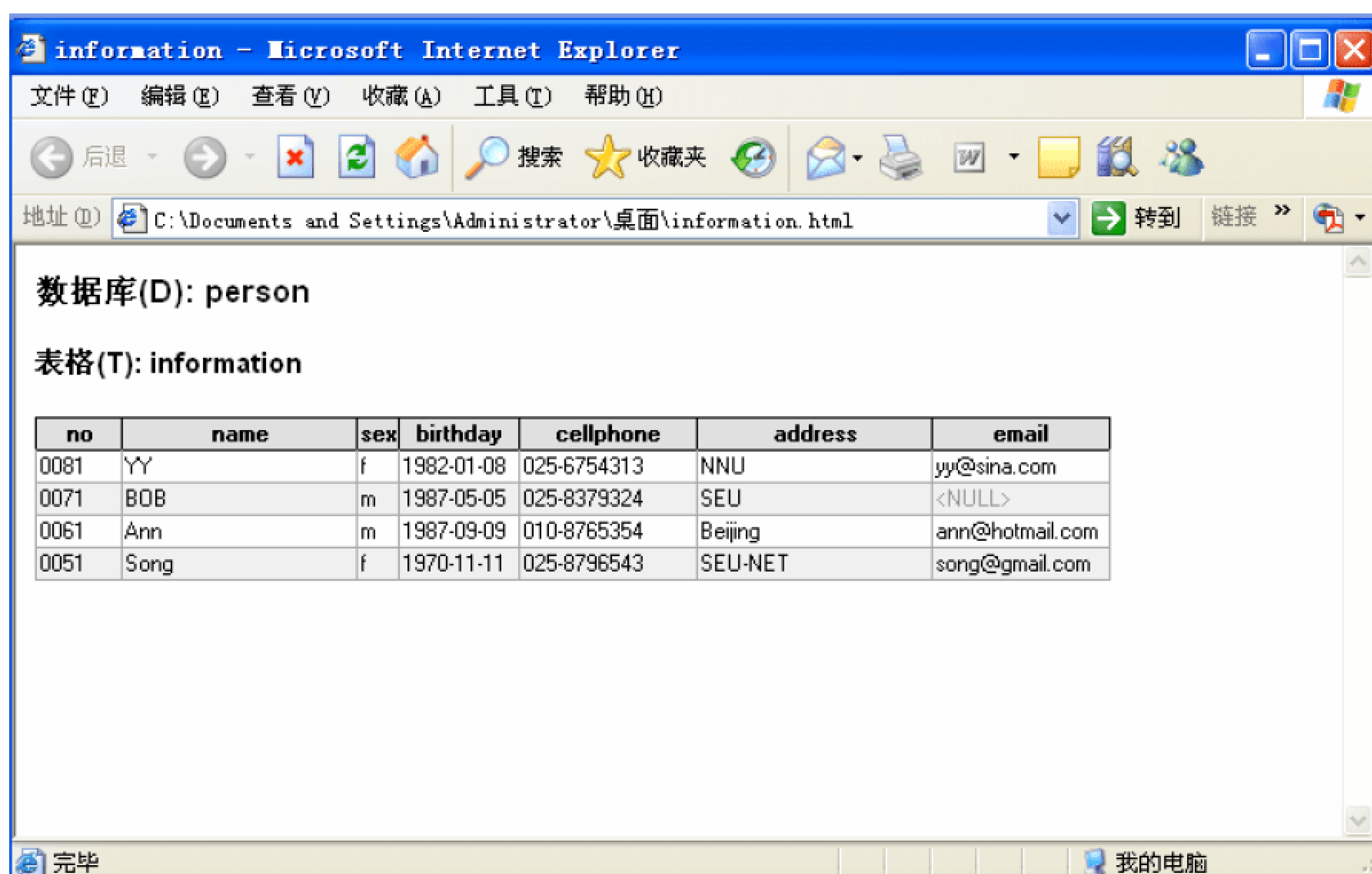


图 11-54　以 HTML 格式输出表数据

📖**点评与拓展：** MySQL-Front 软件提供了图形化管理 MySQL 数据库的方式，大大简化了用户对数据库的操作。

11.4.2　使用 MySQL-Front 实现数据库表的连接操作

❶ 首先添加 course 表，选中 person 数据库右击，在弹出的快捷菜单中选择【新建】→【表格】命令，出现如图 11-55 所示的【添加表格】对话框，输入表的名称为 course；然后切换至【字段】选项卡，单击【新建字段】按钮出现如图 11-56 所示添加新字段的对话框，第一个字段为 stno，类型为 varchar(7)，即最多为 7 个字符的变长串。然后用同样的方法添加第二个字段 coursename1、第三个字段 coursename2，等等，这些 coursename 记录了某学生所选课的名称，可以有多个。

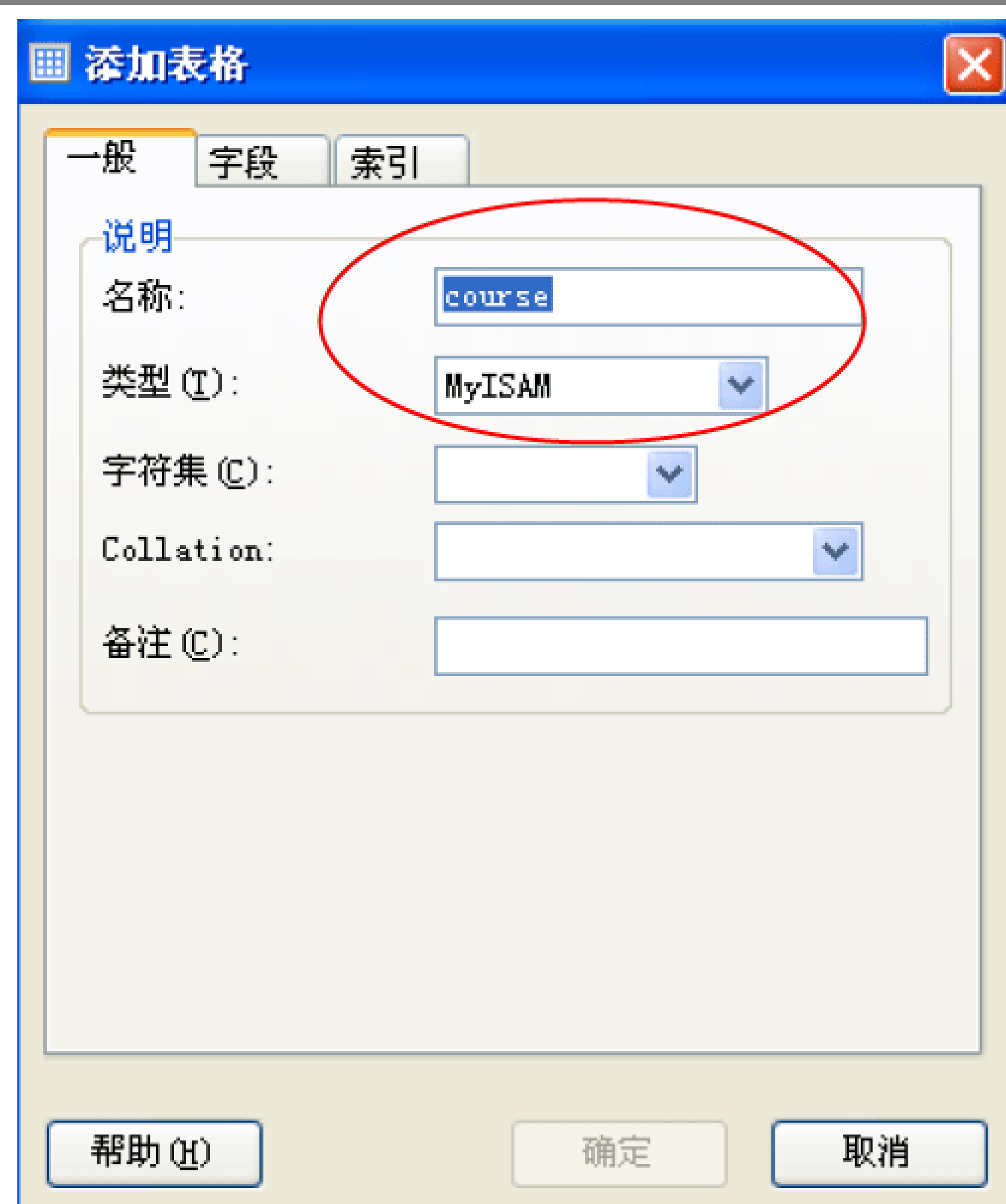


图 11-55　添加 course 表

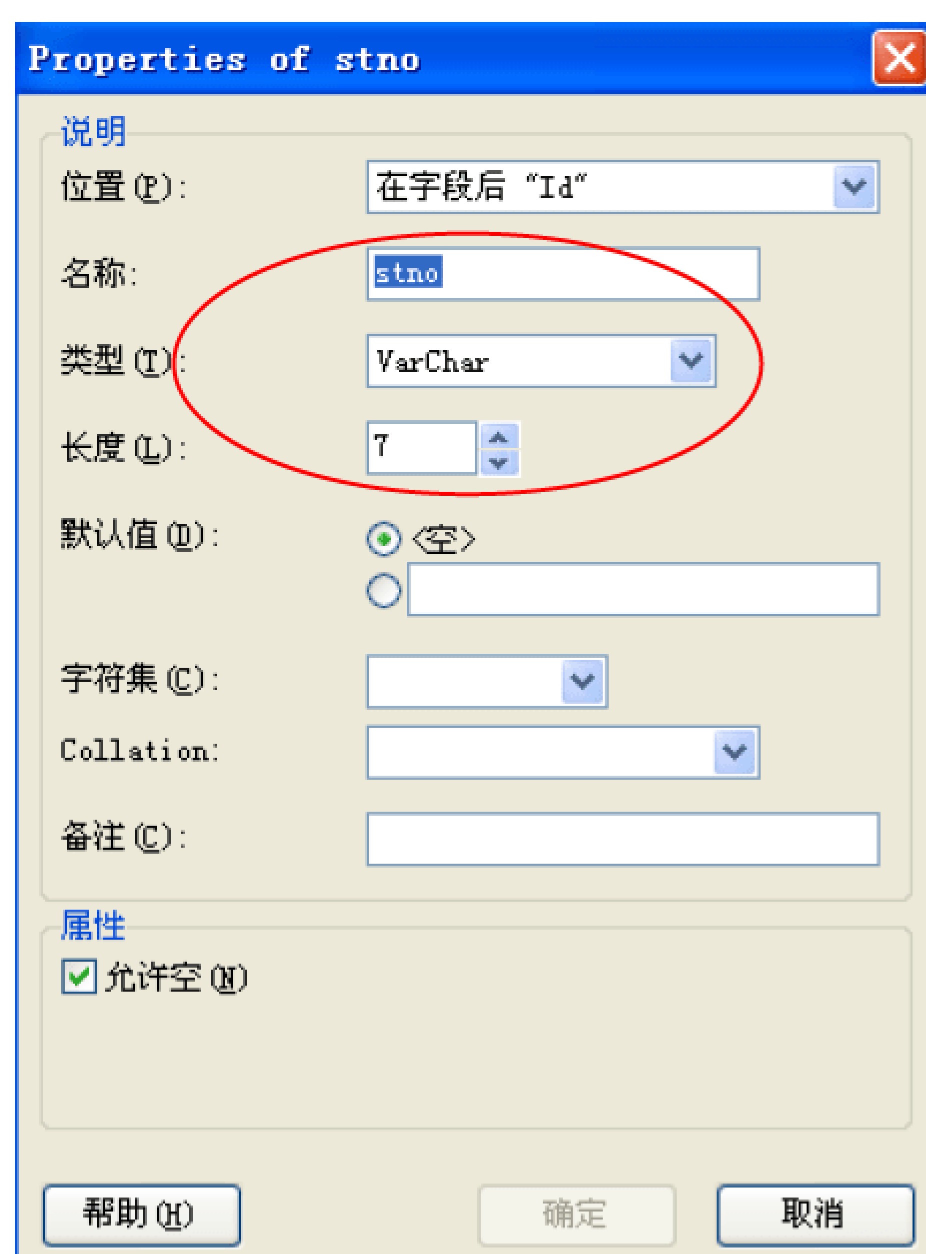


图 11-56　为表 course 添加字段

❷ 创建表 course 的基本字段完毕后，我们为表 course 添加数据，添加数据的方法在上节中介绍过，比如本例我们添加了四个数据，如图 11-57 所示。图 11-58 显示了表 information 的数

据，通过比对这两个表的数据发现，其中的学号为 0071 和 0081 的数据是 information 表中存在记录的学生，另外的两个数据在 information 表中无对应记录。

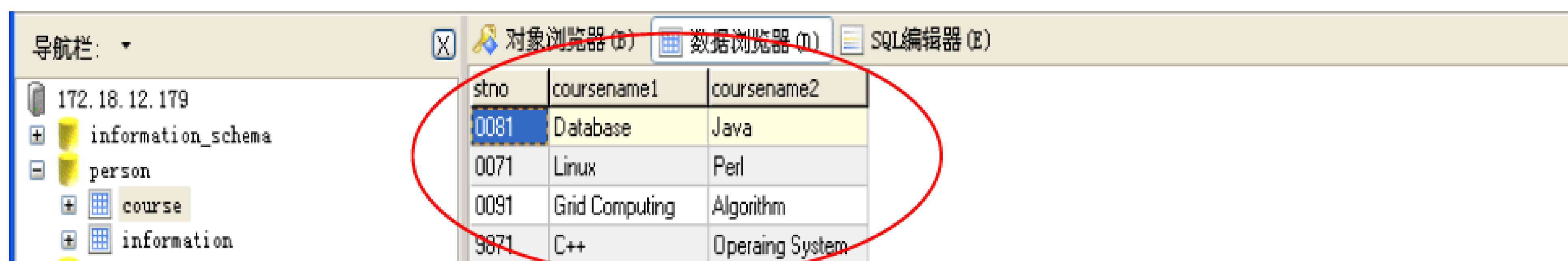


图 11-57　在表 course 添加数据

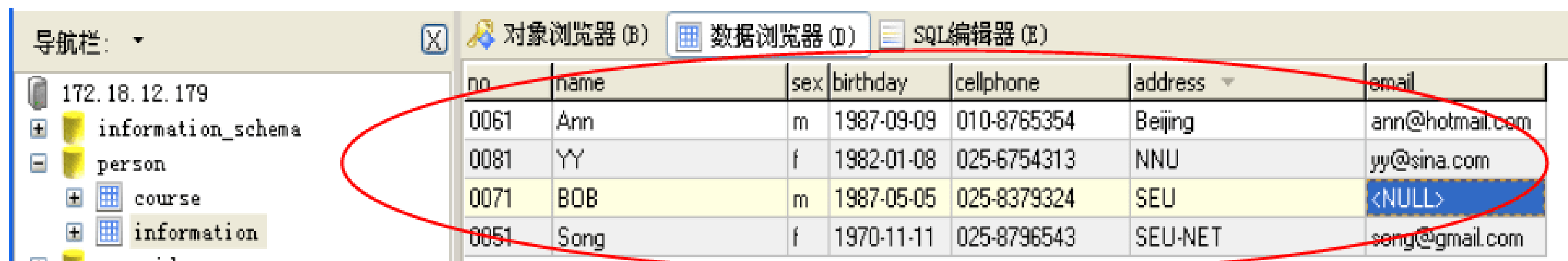


图 11-58　表 information 的数据

❸ 接着我们演示一个简单的数据库表之间的连接，所谓连接，就是查询出存放在多个表中的同一个实体的信息。例如学号为 0081 的学生在 information 表中存放了名字、性别和地址等基本信息，在 course 表中存放了该学生的选课信息，通过连接操作就可以将存储在 information 表和 course 表中的信息一起显示出来，以共用的学号字段为连接纽带。

单击【SQL 编辑器】按钮进入 SQL 语言编辑页面，输入下面的 SQL 语句：

```
select * from information, course
where information.no=course.stno;
```

这个 SQL 语句的意思是，当 information 表中 no 字段与 course 表中的 stno 字段相等时，就将 information 和 course 两个表中的所有字段都选出，SQL 语句以分号结束。

单击上方的三角按钮，即运行 SQL 语句的按钮，如图 11-59 标注部分所示。如果所写入的 SQL 语句有错误，MySQL-Front 将会报错，以帮助用户修改，如图 11-60 所示。

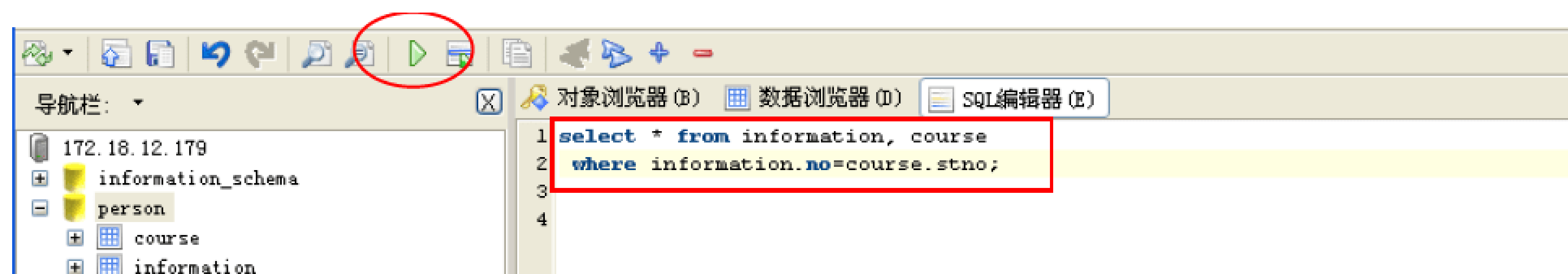


图 11-59　编辑并执行连接 SQL 语句

❹ 执行成功后，在 MySQL-Front 窗口下方跳出连接操作的结果，如图 11-61 标注部分所示，我们可以看到这里列出了学号为 0081 和 0071 的数据记录，而且后面加上了 course 表中的课程信息内容。

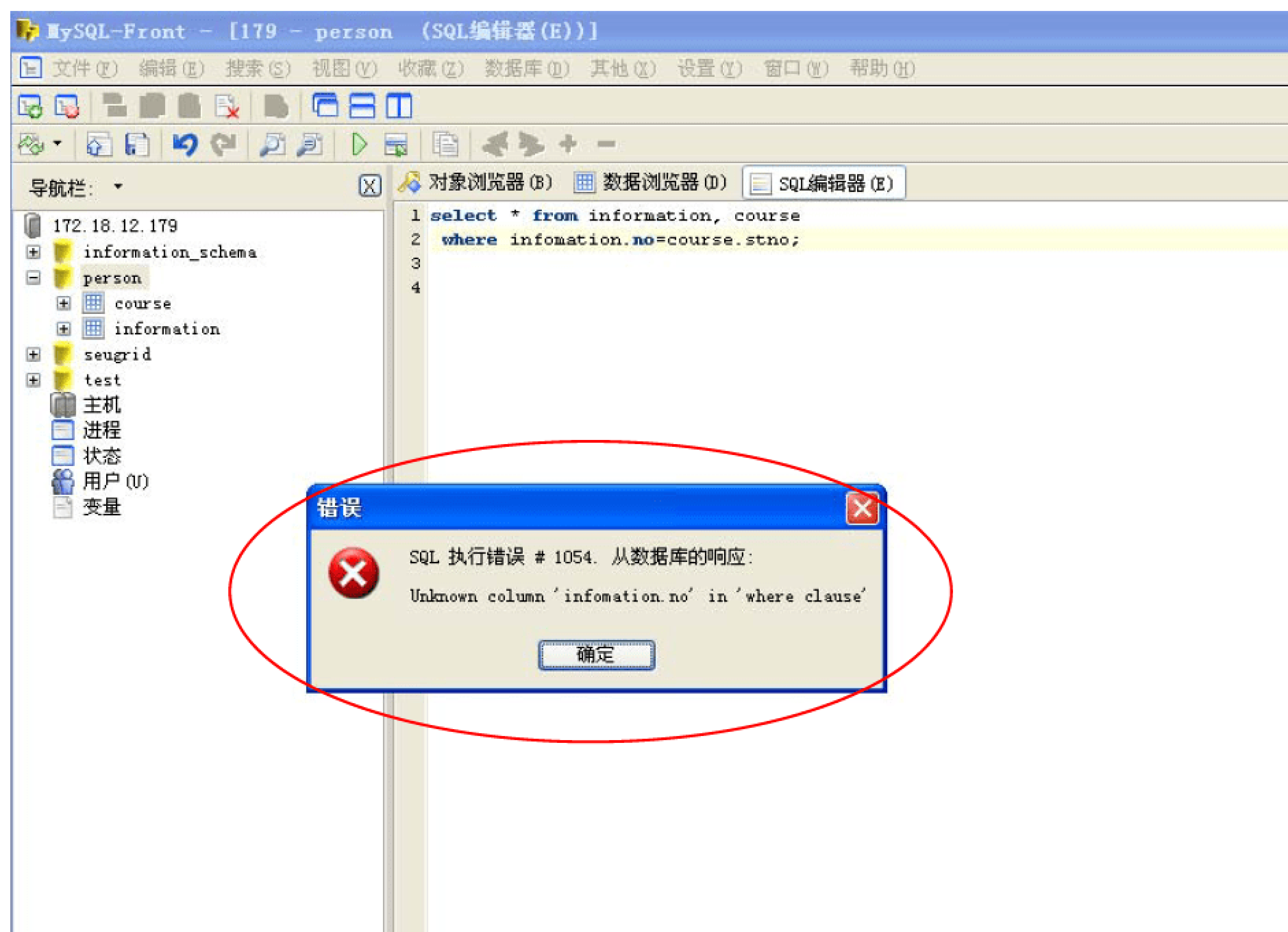


图 11-60　执行 SQL 语句出错

对象浏览器(B)　数据浏览器(D)　SQL编辑器(E)

```
1 select * from information, course
2  where information.no=course.stno;
3
4
```

no	name	sex	birthday	cellphone	address	email	stno	coursename1	coursename2
0081	YY	f	1982-01-08	025-6754313	NNU	<NULL>	0081	Database	Java
0071	BOB	m	1987-05-05	025-8379324	SEU	<NULL>	0071	Linux	Perl

图 11-61　连接的执行结果

📖点评与拓展：　本节仅示例性地利用 MySQL-Front 软件实现了连接操作，它通过批处理 SQL 语言脚本来实现，更多的 MySQL-Front 软件的用法，以及 SQL 语言脚本的用法，有待读者在实践中发掘。

11.5　本 章 小 结

本章主要介绍数据库管理系统、SQL 语言等数据库方面的基础知识，然后介绍流行数据库服务器软件 MySQL 的安装、配置，接着介绍了 MySQL 管理的相关知识，包括如何设定管理员密码、数据库的创建和删除、表的创建和删除、表中数据的插入和修改，以及

MySQL 数据库用户管理的相关知识等。最后介绍如何利用 MySQL-Front 软件图形化的方式来方便地管理 MySQL 服务器，包括 MySQL-Front 软件的连接数据库的设置，以及对数据库的基本操作等，还通过一个简单的例子演示了用 MySQL-Front 软件和 SQL 语言如何实现数据库表的连接操作，希望读者能够通过这个例子举一反三。

第 12 章　LDAP 服务器的配置与架设

LDAP(Light Directory Access Protocol，轻量级目录访问协议)技术是比较新的概念，但它发展迅速，很快成为标准网络体系结构中不可或缺的部分。在企业范围内实现 LDAP，可以让运行在几乎所有计算机平台上的所有应用程序从 LDAP 目录中获取信息，以简化员工在企业内部查询信息的步骤。LDAP 目录中可以分布式存储各种类型的数据：电子邮件地址、邮件路由信息、人力资源数据、公用密匙、联系人列表，等等。本章主要介绍目录服务和 LDAP 的基本概念，重点介绍如何利用 OpenLDAP 软件架设目录服务器，涉及 OpenLDAP 软件的安装、配置和启动，以及如何使用 LDAP Browser/Editor 软件来管理 LDAP 服务器。

通过本章的学习，读者应掌握以下内容：

✧ 目录服务的概念
✧ LDAP 的基本概念和工作原理
✧ OpenLDAP 的安装、配置和启动
✧ 如何使用 LDAP Browser/Editor 软件来管理 LDAP 服务器

12.1　目录服务概述

12.1.1　目录服务简介

随着互联网中服务、资源、计算机的急速增长，为如此多的分布于互联网中的实体进行命名成为了一个非常基础的问题。目录服务正是方便用户阅读、浏览以及搜索的专门数据库，它通过存放名字和描述性的属性的记录为互联网用户提供命名服务、目录服务和发现服务。

显而易见，名字并非是识别互联网实体的惟一途径，描述性的属性也可用于识别实体。比如，一个用户可能会问：“电话号码为 025-83792241”的用户是谁？”又比如，用户需要使用打印机服务，他只要知道哪个服务能够很好地满足其要求，但不关注服务由谁提供，那么这个用户就会问：“我在哪儿可以打印一个高分辨率的彩色图像？”这些实例都说明，在实际的环境中，用户可能会以描述性的属性作为查询值来获得一些信息。因此，目录服务的设计就需要满足这样的需求。而且，目录服务需要针对大量的查找或搜索操作进行快速的响应，它们具有大范围复制信息的功能，以便提高可用性和可靠性，同时缩短响应时间。当目录信息被复制时，副本之间的暂时不一致是允许的，只要最终同步即可。

有许多不同方式可以提供目录服务。不同的方法允许不同类型的信息存储在目录里，

响应不同的请求信息如何被访问、查询和更新，如何避免非特许存取，等等。一些目录服务是局部的，为有限的应用场景提供服务，比如，单机上的手控服务；另一些服务是全局的，为非常广泛的应用场景提供服务，比如，整个 Internet。全局服务通常是分布式的，也就是说全局服务所保存的数据扩展到许多台主机，所有这些主机共同协作提供目录服务。通常，全局服务定义了一个统一的命名空间，互联网域名系统(DNS 服务器)就是一个全局分布式目录服务的例子。

12.1.2　X.500 简介

X.500 是用于目录服务的标准协议，它已经得到了 ITU 和 ISO 标准组织的官方认可，称 X.500 是一个访问有关“现实世界实体”信息的服务。

从技术上来说，LDAP 是 X.500 目录服务的一个目录访问协议，最初，LDAP 客户端访问 X.500 目录服务的网关，该网关在客户端和网关之间运行 LDAP，在网关和 X.500 服务器之间运行 X.500 的目录访问协议.DAP(重量级协议)，运行全部 OSI 协议组需要相当大的计算资源。于是，LDAP 设计为运行在 TCP/IP 协议上，以非常低的开销提供 DAP 的大部分功能。现在，LDAP 更普遍地在 X.500 服务器上直接执行，尽管仍要通过网关去访问 X.500 目录服务。

独立的 LDAP 进程可以看作是一个轻量级 X.500 目录服务器。也就是说，它既不执行 X.500 的目录访问协议.DAP，也不支持完整的 X.500 模型。

12.1.3　主要目录服务产品简介

1．OpenLDAP

OpenLDAP 是一款出色的开放源代码 LDAP 服务器软件，其目标是提供一个稳定的、商业化的、功能全面的 LDAP 软件，包括 LDAP 服务器和一些开发工具。由于 OpenLDAP 是开放源代码的，因此它在 UNIX 和 Linux 平台上受到广泛的欢迎。当然它也可以移植到其他的系统平台，甚至是 Windows 平台上，从而满足用户的不同需求。OpenLDAP 2.x 版本支持 LDAP v3，而最新的版本(2.2 版)可以支持 LDAP v3 协议的绝大部分特性，包括一些扩展功能。OpenLDAP 不但功能强大而且安全可靠，目前被许多大型的邮件系统和 ISP 所使用。

2．Microsoft Active Directory(活动目录)

活动目录是 Windows 2000 和 Windows 2003 的核心组件，它利用目录服务功能，将整个网络系统有机地组织起来。目前活动目录成了 Windows 2000 和 Windows 2003 网络系统的核心，它存储了当前网络环境中所有资源的信息，包括基本个人账户信息和各种系统服务。另外，活动目录本身与系统的安全服务紧密地集成在一起，每个用户的安全信息被保存在活动目录中，而用户对系统资源的访问也受活动目录控制。而且操作系统通过活动目录控制用户的登录，因为活动目录与 Kerberos 认证协议结合起来，实现了单点登录(single sign-on)特性。

Microsoft 的活动目录是一个全面的目录服务管理方案，也是一个企业级的目录服务，具有很好的可扩展性。它采用了 Internet 的标准协议，并与操作系统紧密地集成在一起，不仅可以管理基本的网络资源，比如计算机对象、用户账户、打印机等，还充分考虑了现代应用的业务需求，为这些应用提供了基本的管理对象模型，比如用户账户对象具有办公电话、手机、呼机、住址、上司、下属和电子邮件等属性。几乎所有的应用都可以直接利用系统提供的目录服务功能，而且活动目录也具有很好的扩展能力，允许应用程序定制目录中对象的属性或者添加新的对象类型。

3．NDS

NDS(Novell Directory Services)是随着 Netware 4 一起发布的，属于比较早的面向企业网络的目录服务产品。它有效地将网络系统的各种资源组织到一起，也将各种应用软件集成到同一个资源管理平台上。NDS 可以支持各种规模的网络环境，当然也支持 LDAP v3。由于 NDS 是从系统软件中剥离出来的，因此软件的可靠性较高，也支持多个目录服务器之间的复制。

到了 Netware 5 的时候，NDS 从中分离出来，以便支持不同的系统平台，这就是 Novell eDirectory。Novell eDirectory 是一种支持 LDAP，基于目录的身分管理系统，它对用户身份、访问特权和其他网络资源实行集中管理。由于 eDirectory 支持动态组功能，能够持久搜索和实时连续备份，因此大大降低了管理费用。eDirectory 卓越的安全功能包括“Novell 国际密码基础结构”、通过安全套接层加密的口令、RSA 私用密钥加密、安全性鉴定服务、智能卡和 X.509v3 证书。管理员授予一个目录的访问权限，并不意味着授予对整个网络的访问权，甚至也不表明可以访问该目录中的全部信息，从而可以确切地指定哪些人可以访问哪些信息。

12.2　LDAP 简介

12.2.1　LDAP 概念

X.500 协议虽然功能强大，但实现起来非常繁琐，效率不高，往往在许多小系统中无法实现。为了降低复杂性和开发成本，IETF(Internet Engineering Task Force)制定了轻量级目录访问协议(LDAP)。

轻量级目录访问协议(LDAP)是一个独立于厂家和平台的开放网络协议标准，它是用来访问存储在信息目录中的信息协议，更为确切的说法应该是：通过使用 LDAP，可以在信息目录的正确位置读取数据。它在对 X.500 标准进行简化的基础上，基于 TCP/IP 定义了一个目录服务标准，主要包括以下几个部分。

- ✧ LDAP 信息模型：定义了目录中数据的类型；
- ✧ LDAP 命名模型：定义了目录的组织方式；
- ✧ LDAP 函数模型：定义了如何访问和更新目录；

- ✧ LDAP 安全模型：定义了如何防止未经授权的用户对目录信息的访问和修改。

在 LDAP 目录中信息是存储在一个树型结构中的，一般称为 DIT(目录信息树)。DIT 由许多记录项组成。记录项表示 LDAP 中的资源信息，每个记录项具有惟一的标识，并包含零个或者多个属性值，表示资源的属性。记录项的类型，称为对象类，确定了记录项和属性的必选项和可选项。就像 SYBASE、ORA-CLE、INFORMIX 或者 MICROSOFT 的数据库管理系统是用于处理查询和更新关系型数据库那样，LDAP 服务器也是用来处理查询和更新 LDAP 目录的。换句话说，LDAP 目录也是一种类型数据库，而不是关系数据库。

LDAP 主要优化了数据读取的性能，适合于更新频率远小于读取频率的情况。LDAP 和普通数据库在实现上有差别，在性能上也有所不同。它的主要优点包括：

- ✧ LDAP 是跨平台的协议，可以在任何平台的计算机上，用 LDAP 客户端软件去访问 LDAP 服务器；
- ✧ 对 LDAP 的读操作的完成速度比普通的数据库访问要快得多，LDAP 专门为读操作进行了优化，适合于需要频繁读取的场合；
- ✧ LDAP 的服务器也可以是分布式的，用户访问到的信息可以是本地的 LDAP 服务器，也可以是各地的服务器。LDAP 可以通过内部机制很容易地实现各地服务器之间内容的同步；
- ✧ LDAP 的存储是按照一条条记录项进行的，各条记录项存储的属性是可变的。

完成资源的 LDAP 定义和表示后，就需要根据资源的表示方法对资源信息进行有效的组织。根据资源分类的特性，使用层次化的树型结构对资源信息进行组织是比较合理的，这样便于系统对资源信息的查找、添加等操作。资源信息的组织需要遵循：

- ✧ 资源的标识应保证这个资源在系统中的惟一性，不能出现多个资源标识指向同一个资源或一个资源标识指向多个资源的情况；
- ✧ 资源分类应具有多个层次，同时树型组织结构应清楚地体现不同模型的差异；
- ✧ 资源的属性可以冗余，保证资源的完整表示。

资源目录管理系统对信息资源采用了多层次的树型组织方式。在树型结构中，每个结点代表一个对象类，每个对象类中都定义了其父结点和子结点。每个对象类都可以对应多个记录项。同时，这种树型的结构便于资源信息的查找，可提供高效的资源定位方式。

在 LDAP 协议中存在两种通信模式：客户-服务器通信和服务器－服务器通信。基本的客户-服务器通信允许用户程序连接 LDAP 服务器进行创建、检索、修改、删除数据等操作。服务器-服务器通信定义了多个服务器如何共享一个 LDAP 目录信息树(DIT)，以及如何更新和复制服务器之间的信息。

12.2.2 LDAP 基本原理

LDAP 是一个访问目录服务的轻量级协议，特别是基于 X.500 的目录服务。它运行在 TCP/IP 上或面向传输服务的其他连接。LDAP 相关的规定在 RFC2251 和 RFC3377 等文档里有详细说明。

LDAP 信息模型是基于记录项的，一个记录项是一个属性的集合，它有一个全局惟一的标识名(DN)。每个条目属性都对应于类型(type)和值(value)的组合，类型通常是助记符，例

如 cn 是公共名称，mail 是 E-mail 地址。值的语法取决于属性类型，例如，cn 属性可以包含值 Babs Jensen；mail 属性可以包含值babs@example.com；jpegPhoto 属性可以包含 JPEG 格式的图片；等等。LDAP 通过使用一个称为 objectClass 属性的值来决定记录项必须遵守的模式规则，而且 LDAP 可以控制一个记录项中哪些属性是必选的，哪些属性是可选的。

那么信息是如何组织的呢？简单地说，LDAP 的记录项是以一个层次树结构组织起来的，下面分别介绍两种命名方式：传统命名法和互联网域命名法。

1．传统命名法

传统的层次树结构反映了地理或组织的分界。树顶是代表国家的记录项，接下来是代表州和省的记录项，再接下来可以是代表城市的记录项，最后可以是组织单位、人、打印机、文件或你所能想到的任何事物的记录项。图 12-1 给出了一个 LDAP 目录树的例子，它使用了传统命名法。

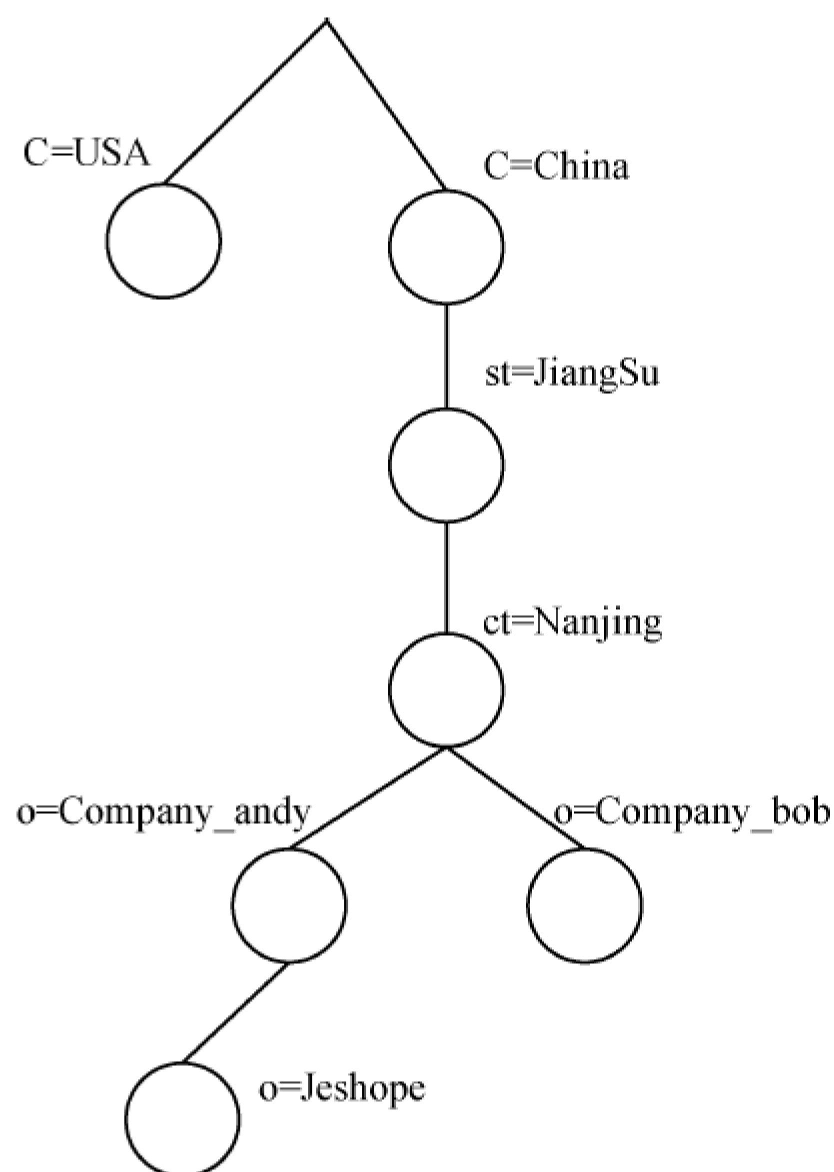


图 12-1　传统命名法例子

2．互联网域命名法

第 7 章“DNS 服务器的配置与架设”中介绍了 DNS 域名的组织方式，这种组织方式也是层次式的，LDAP 借鉴 DNS 域名的组织方式，提出了互联网域名命名法，而且该方法变得越来越流行。图 12-2 给出了一个使用域名命名法的 LDAP 目录树例子。一个记录项通过标识名(DN)被访问，在获得记录项自身的相对标识名之后，将其和它祖先记录项名连接，就构成了标识名(DN)。例如，该例子里有一个 uid=Jeshope 的 RDN，将其与它所有祖先连接后，形成的完整的表示名就为：Jeshope.people.example.com，这个过程正如我们第 7 章所讲述的反向解析 DNS 域名一样。

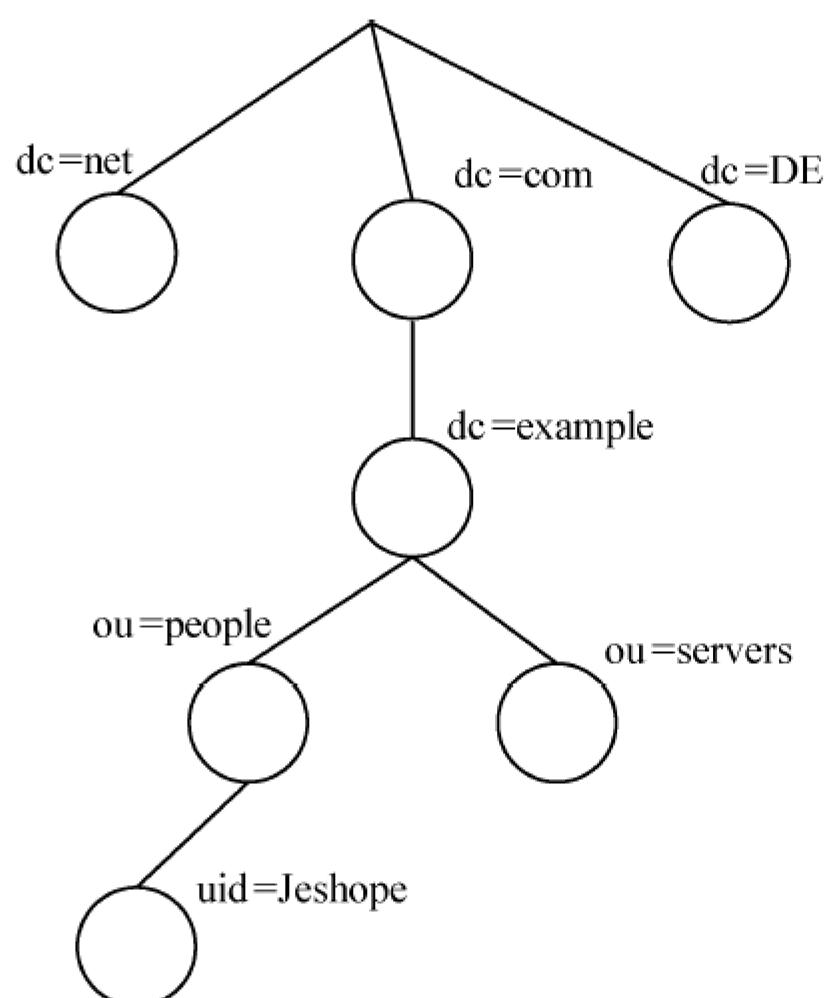


图 12-2　互联网域命名法例子

12.2.3　LDAP 的应用领域

目录并不是数据库，但它们有助于网络基础结构的管理。就像一本好书需要详细的目录来帮助读者查找所需的页面一样，网络通过目录服务也可使用户对它的任一部分进行快速的访问。因此，LDAP 具有查询效率高、树状的信息管理模式、分布式部署框架以及灵活细腻的访问控制等特点，这使得 LDAP 广泛应用于基础性、关键性信息的管理。LDAP 的应用主要涉及以下几种类型。

- ✧ 信息安全类：数字证书管理、授权管理、单点登录。
- ✧ 科学计算类：DCE(分布式计算环境)，UDDI(统一描述、发现及集成协议)。
- ✧ 网络资源管理类：MAIL 系统、DNS 系统、网络用户管理、电话号码簿。
- ✧ 电子政务资源管理类：内网组织信息服务、电子政务目录体系；人口基础库、法人基础库。

12.3　LDAP 的安装

应用实例导航——为 A 公司安装 LDAP 服务器

※场景呈现

A 公司需要架设目录服务器，选用 OpenLDAP 软件；试为 A 公司安装 Berkerley 数据库

和 OpenLDAP 软件，并通过加载.ldif 文件初始化 OpenLDAP 中的目录服务。

※技术要领

(1) Berkerley 数据库的安装。
(2) OpenLDAP 的安装。
(3) OpenLDAP 的初始化。

OpenLDAP 是架设 LDAP 服务器的常用软件，本章就是利用它来架设 LDAP 服务器的。首先涉及的就是 OpenLDAP 的安装问题，本节介绍如何安装、配置和启动 OpenLDAP。

12.3.1　Berkerley 数据库的安装

在安装 OpenLDAP 之前，首先要安装 Berkerley 数据库(BDB)，因为 Berkerley 数据库是 OpenLDAP 的前驱软件，即它们之间存在依赖关系。OpenLDAP 后台数据库用的是 BDB4.2。

1. 本节演示的 Berkerley 数据库的版本是 4.2.52，通过下载或者从本书附带光盘都可以获得 Berkerley 数据库的安装包，名字为 db-4.2.52.NC.tar.gz，我们使用 tar 命令将安装包解压到 /usr/local 目录下，如图 12-3 所示。

```
tar -zxvf db-4.2.52.NC.tar.gz
```

```
[root@seugrid3 local]# tar -zxvf db-4.2.52.NC.tar.gz
db-4.2.52.NC/
db-4.2.52.NC/LICENSE
db-4.2.52.NC/README
db-4.2.52.NC/btree/
db-4.2.52.NC/btree/bt_compare.c
db-4.2.52.NC/btree/bt_conv.c
db-4.2.52.NC/btree/bt_curadj.c
db-4.2.52.NC/btree/bt_cursor.c
```

图 12-3　解压 Berkerley 数据库安装包

2. 解压后，在/usr/local 下生成 db-4.2.52.NC 目录。进入 db-4.2.52.NC/build_UNIX 目录，我们使用下面命令可以看到有个名字为“.”的隐藏目录。

```
ls -a
```

ls 是列出目录内容命令，-a 参数列出目录下的所有文件及目录。Linux 文件系统中，以“.”开头的文件或文件夹都为隐藏文件，如第 8 章所讲到的配置环境变量的.bash_profile 文件，所以如果不带-a 参数将无法显示这些隐藏文件或文件夹。
配置文件 configure 就在此目录的 dist 子目录下，因此输入下面命令进行配置，两条命令的运行结果如图 12-4 所示。

```
../dist/configure
```

请注意，上面命令两个“.”之间没有空格，而且这里的 configure 不需要带-prefix 参数，Berkerley 数据库默认安装在/usr/local 目录下。

```
[root@seugrid3 db-4.2.52.NC]# cd build_unix
[root@seugrid3 build_unix]# ls -a
.  ..  .IGNORE_ME  tags
[root@seugrid3 build_unix]# ../dist/configure
checking build system type... i686-pc-linux-gnu
checking host system type... i686-pc-linux-gnu
checking if building in the top-level or dist directories... no
checking if --disable-cryptography option specified... no
checking if --disable-hash option specified... no
checking if --disable-queue option specified... no
checking if --disable-replication option specified... no
checking if --disable-verify option specified... no
```

图 12-4　配置 Berkerley 数据库安装包

❸ 配置完毕后，接下来就是编译 Berkerley 数据库安装包，注意，还是在 build_UNIX 目录下输入下面命令，如图 12-5 所示。

```
make
```

```
[root@seugrid3 build_unix]# make
/bin/sh ./libtool --mode=compile cc -c -I. -I../dist/..  -D_GNU_SOURCE -D_REENTRANT -O2 ../dist/../mutex/mut_pthread.c
mkdir .libs
 cc -c -I. -I../dist/.. -D_GNU_SOURCE -D_REENTRANT -O2 ../dist/../mutex/mut_pthread.c  -fPIC -DPIC -o .libs/mut_pthread.o
 cc -c -I. -I../dist/.. -D_GNU_SOURCE -D_REENTRANT -O2 ../dist/../mutex/mut_pthread.c -o mut_pthread.o >/dev/null 2>&1
/bin/sh ./libtool --mode=compile cc -c -I. -I../dist/..  -D_GNU_SOURCE -D_REENTRANT -O2 ../dist/../btree/bt_compare.c
 cc -c -I. -I../dist/.. -D_GNU_SOURCE -D_REENTRANT -O2 ../dist/../btree/bt_compare.c  -fPIC -DPIC -o .libs/bt_compare.o
 cc -c -I. -I../dist/.. -D_GNU_SOURCE -D_REENTRANT -O2 ../dist/../btree/bt_compare.c -o bt_compare.o >/dev/null 2>&1
/bin/sh ./libtool --mode=compile cc -c -I. -I../dist/..  -D_GNU_SOURCE -D_REENTRANT -O2 ../dist/../btree/bt_conv.c
 cc -c -I. -I../dist/.. -D_GNU_SOURCE -D_REENTRANT -O2 ../dist/../btree/bt_conv.c  -fPIC -DPIC -o .libs/bt_conv.o
 cc -c -I. -I../dist/.. -D_GNU_SOURCE -D_REENTRANT -O2 ../dist/../btree/bt_conv.c -o bt_conv.o >/dev/null 2>&1
/bin/sh ./libtool --mode=compile cc -c -I. -I../dist/..  -D_GNU_SOURCE -D_REENTRANT -O2 ../dist/../btree/bt_curadj.c
 cc -c -I. -I../dist/.. -D_GNU_SOURCE -D_REENTRANT -O2 ../dist/../btree/bt_curadj.c  -fPIC -DPIC -o .libs/bt_curadj.o
 cc -c -I. -I../dist/.. -D_GNU_SOURCE -D_REENTRANT -O2 ../dist/../btree/bt_curadj.c -o bt_curadj.o >/dev/null 2>&1
/bin/sh ./libtool --mode=compile cc -c -I. -I../dist/..  -D_GNU_SOURCE -D_REENTRANT -O2 ../dist/../btree/bt_cursor.c
 cc -c -I. -I../dist/.. -D_GNU_SOURCE -D_REENTRANT -O2 ../dist/../btree/bt_cursor.c  -fPIC -DPIC -o .libs/bt_cursor.o
 cc -c -I. -I../dist/.. -D_GNU_SOURCE -D_REENTRANT -O2 ../dist/../btree/bt_cursor.c -o bt_cursor.o >/dev/null 2>&1
```

图 12-5　编译 Berkerley 数据库安装包

❹ 编译无误后，就可以安装 Berkerley 数据库安装包了，在 build_UNIX 目录下输入下面命令，如图 12-6 所示。

```
make install
```

📖**点评与拓展：**安装 Berkerley 数据库的特殊之处在于，配置、编译和安装命令都需要在 build_UNIX 目录下输入。

```
[root@seugrid3 build_unix]# make install
Installing DB include files: /usr/local/BerkeleyDB.4.2/include ...
Installing DB library: /usr/local/BerkeleyDB.4.2/lib ...
cp -p .libs/libdb-4.2.so /usr/local/BerkeleyDB.4.2/lib/libdb-4.2.so
cp -p .libs/libdb-4.2.lai /usr/local/BerkeleyDB.4.2/lib/libdb-4.2.la
cp -p .libs/libdb-4.2.a /usr/local/BerkeleyDB.4.2/lib/libdb-4.2.a
ranlib /usr/local/BerkeleyDB.4.2/lib/libdb-4.2.a
chmod 644 /usr/local/BerkeleyDB.4.2/lib/libdb-4.2.a
cp -p libdb.a /usr/local/BerkeleyDB.4.2/lib/libdb.a
ranlib /usr/local/BerkeleyDB.4.2/lib/libdb.a
chmod 644 /usr/local/BerkeleyDB.4.2/lib/libdb.a
PATH="$PATH:/sbin" ldconfig -n /usr/local/BerkeleyDB.4.2/lib
----------------------------------------------------------------------
Libraries have been installed in:
   /usr/local/BerkeleyDB.4.2/lib

If you ever happen to want to link against installed libraries
in a given directory, LIBDIR, you must either use libtool, and
specify the full pathname of the library, or use the `-LLIBDIR'
flag during linking and do at least one of the following:
   - add LIBDIR to the `LD_LIBRARY_PATH' environment variable
     during execution
   - add LIBDIR to the `LD_RUN_PATH' environment variable
     during linking
   - use the `-Wl,--rpath -Wl,LIBDIR' linker flag
   - have your system administrator add LIBDIR to `/etc/ld.so.conf'

See any operating system documentation about shared libraries for
more information, such as the ld(1) and ld.so(8) manual pages.
----------------------------------------------------------------------
Installing DB utilities: /usr/local/BerkeleyDB.4.2/bin ...
cp -p .libs/db_archive /usr/local/BerkeleyDB.4.2/bin/db_archive
cp -p .libs/db_checkpoint /usr/local/BerkeleyDB.4.2/bin/db_checkpoint
cp -p .libs/db_deadlock /usr/local/BerkeleyDB.4.2/bin/db_deadlock
cp -p .libs/db_dump /usr/local/BerkeleyDB.4.2/bin/db_dump
cp -p .libs/db_load /usr/local/BerkeleyDB.4.2/bin/db_load
cp -p .libs/db_printlog /usr/local/BerkeleyDB.4.2/bin/db_printlog
cp -p .libs/db_recover /usr/local/BerkeleyDB.4.2/bin/db_recover
cp -p .libs/db_stat /usr/local/BerkeleyDB.4.2/bin/db_stat
cp -p .libs/db_upgrade /usr/local/BerkeleyDB.4.2/bin/db_upgrade
cp -p .libs/db_verify /usr/local/BerkeleyDB.4.2/bin/db_verify
Installing documentation: /usr/local/BerkeleyDB.4.2/docs ...
[root@seugrid3 build_unix]#
```

图 12-6　安装 Berkerley 数据库安装包

12.3.2　OpenLDAP 的安装

OpenLDAP 的安装除了需要 Berkerley 数据库的支持外，还需要很多第三方软件的支持，比如 C 语言编译器 gcc 等。不过一般情况下，其他的第三方软件在 Linux 安装时就已经安装完毕了。本节假设在其他软件都安装完成的前提下，直接开始安装 OpenLDAP。

OpenLDAP 是免费软件，可以到官方网站http://www.openldap.org下载，如图 12-7 所示。这个网站不仅提供了 OpenLDAP 软件免费下载，还提供了 OpenLDAP 的安装、管理的相关文档，以及常见问题的解答(FAQ)。值得推荐的是http://www.openldap.org/faq/data/cache/53.html，如图 12-8 所示，这里记录了使用 OpenLDAP 时可能碰到的错误类型及其原因，当用户在安装或使用 OpenLDAP 遇到错误时，在此网页查询错误类型，就能找到相应的原因，这对初学者十分有用。

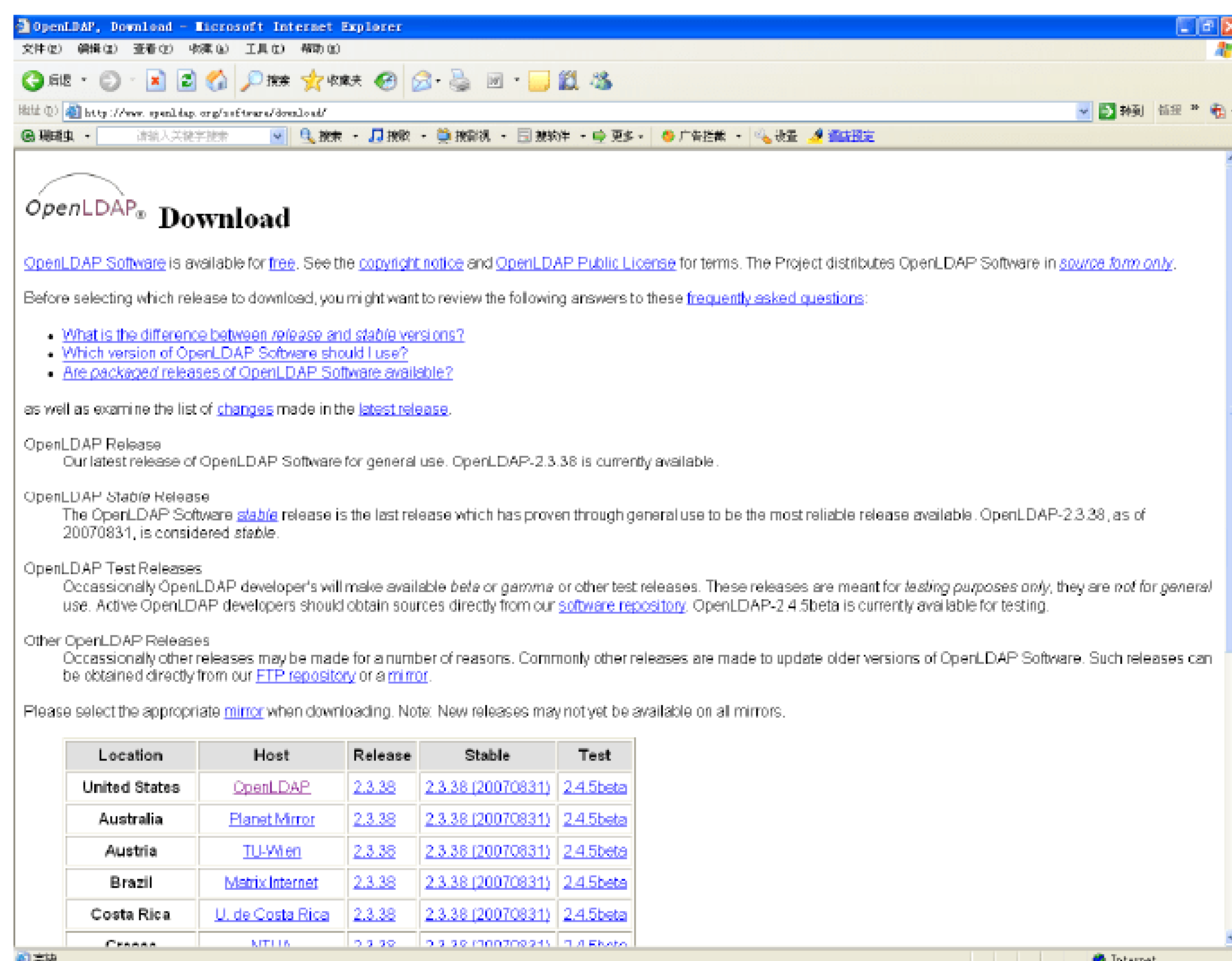


图 12-7　下载 OpenLDAP

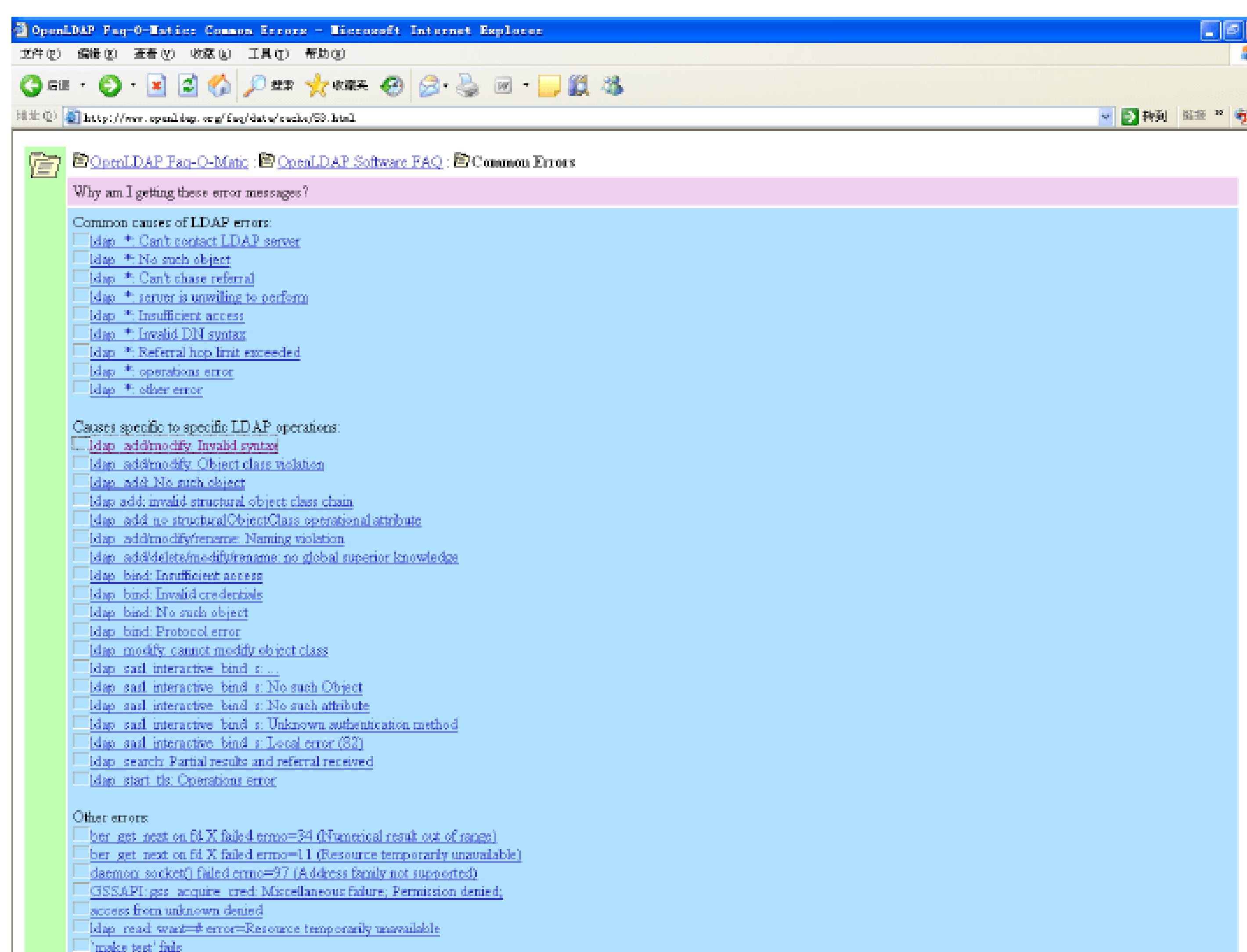


图 12-8　使用 OpenLDAP 时的错误类型

❶ 将下载的 OpenLDAP 安装包复制到/usr/local 目录，本节以名字为 openldap-2.3.38.tgz 的安装包为例进行介绍，使用下面命令解压缩该安装包，如图 12-9 所示。解压成功后生成/usr/local/openldap-2.3.38 目录。

```
tar -zxvf openldap-2.3.38.tgz
```

```
[root@seugrid3 local]# tar -zxvf openldap-2.3.38.tgz
openldap-2.3.38
openldap-2.3.38/doc
openldap-2.3.38/ANNOUNCEMENT
openldap-2.3.38/CHANGES
openldap-2.3.38/COPYRIGHT
openldap-2.3.38/INSTALL
openldap-2.3.38/LICENSE
openldap-2.3.38/Makefile.in
openldap-2.3.38/README
```

图 12-9　解压缩 OpenLDAP 安装包

❷ OpenLDAP 在编译过程中需要 Berkeley DB 4.2 的头文件和库，具体指的是目录/usr/local/BerkeleyDB4.2/include 和/usr/local/BerkeleyDB4.2/lib 中的所有文件。在确定 BerkeleyDB4.2 已经安装的情况下，使用下面命令将 BerkeleyDB4.2 的头文件和库复制到 Linux 系统的头文件和库文件目录下，如图 12-10 所示。

```
cp -f /usr/local/BerkeleyDB4.2/include/* /usr/include
cp -f /usr/local/BerkeleyDB4.2/lib/* /usr/lib
```

/usr/include 和/usr/lib 两个目录是系统路径里设定寻找头文件和库的目录，将 BerkeleyDB 的头文件和库文件复制到其中，编译 OpenLDAP 时就能自动找到。输入以上两个命令后，shell 提示是否覆盖一些同名文件，输入 yes 即可。

```
[root@seugrid3 local]# cd BerkeleyDB.4.2/
[root@seugrid3 BerkeleyDB.4.2]# cp -f include/* /usr/include/
cp: 是否覆盖"/usr/include/db_cxx.h"? y
cp: 是否覆盖"/usr/include/db.h"? y
[root@seugrid3 BerkeleyDB.4.2]# cp -f lib/* /usr/lib/
cp: 是否覆盖"/usr/lib/libdb.so"? y
[root@seugrid3 BerkeleyDB.4.2]#
```

图 12-10　复制 Berkeley 库文件

❸ 接着，进入 OpenLDAP 的安装目录，输入下面命令进行安装配置，如图 12-11 所示。

```
./configure --prefix=/usr/local/openldap --without-cyrus-sasl
```

配置参数的说明如下:

--prefix=/usr/local/openldap 指定了 OpenLDAP 的安装目录为/usr/local/openldap;

--without-cyrus-sasl 表示 OpenLDAP 在客户端和服务器端不需要提供 Cyrus 的 SASL(simple authentication and securityl)服务。

```
[root@seugrid3 local]# cd openldap-2.3.38
[root@seugrid3 openldap-2.3.38]# ./configure --prefix=/usr/local/openldap --without-cyrus-sasl
Configuring OpenLDAP 2.3.38-Release ...
checking build system type... i686-pc-linux-gnu
checking host system type... i686-pc-linux-gnu
checking target system type... i686-pc-linux-gnu
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for gawk... gawk
```

图 12-11　配置 OpenLDAP 安装包

❹ 完成 OpenLDAP 安装包的安装配置后，使用下面命令建立软件与库文件的所有依赖关系，如图 12-12 所示。

```
make depend
```

```
[root@seugrid3 openldap-2.3.38]# make depend
Making depend in /usr/local/openldap-2.3.38
  Entering subdirectory include
make[1]: Entering directory `/usr/local/openldap-2.3.38/include'
Making ldap_config.h
make[1]: Leaving directory `/usr/local/openldap-2.3.38/include'

  Entering subdirectory libraries
make[1]: Entering directory `/usr/local/openldap-2.3.38/libraries'
Making depend in /usr/local/openldap-2.3.38/libraries
```

图 12-12　建立 OpenLDAP 安装包的库文件依赖关系

❺ 编译 OpenLDAP 安装包和编译其他安装包一样，输入下面命令即可，如图 12-13 所示。

```
make
```

```
[root@seugrid3 openldap-2.3.38]# make
Making all in /usr/local/openldap-2.3.38
  Entering subdirectory include
make[1]: Entering directory `/usr/local/openldap-2.3.38/include'
make[1]: Nothing to be done for `all'.
make[1]: Leaving directory `/usr/local/openldap-2.3.38/include'

  Entering subdirectory libraries
make[1]: Entering directory `/usr/local/openldap-2.3.38/libraries'
Making all in /usr/local/openldap-2.3.38/libraries
  Entering subdirectory liblutil
make[2]: Entering directory `/usr/local/openldap-2.3.38/libraries/liblutil'
rm -f version.c
```

图 12-13　编译 OpenLDAP 安装包

❻ 编译完 OpenLDAP 安装包后，就可以输入下面命令测试。这些测试包括启动和停止服务器，加载、搜索和修改数据，最后是服务器复制，如图 12-14 所示。这些测试将近有 45 项，需要很长时间，应耐心等待。

```
make test
```

```
[root@seugrid3 openldap-2.3.38]# make test
cd tests; make test
make[1]: Entering directory `/usr/local/openldap-2.3.38/tests'
make[2]: Entering directory `/usr/local/openldap-2.3.38/tests'
Initiating LDAP tests for BDB...
Running ./scripts/all...
>>>>> Executing all LDAP tests for bdb
>>>>> Starting test000-rootdse ...
running defines.sh
Starting slapd on TCP/IP port 9011...
Using ldapsearch to retrieve the root DSE...
Using ldapsearch to retrieve the cn=Subschema...
Using ldapsearch to retrieve the cn=Monitor...
```

图 12-14　测试 OpenLDAP 安装包

❼ 最后一步就是安装 OpenLDAP 了，输入下面命令进行安装，如图 12-15 所示。

```
make install
```

```
[root@seugrid3 openldap-2.3.38]# make install
Making all in /usr/local/openldap-2.3.38
  Entering subdirectory include
make[1]: Entering directory `/usr/local/openldap-2.3.38/include'
make[1]: Nothing to be done for `all'.
make[1]: Leaving directory `/usr/local/openldap-2.3.38/include'

  Entering subdirectory libraries
make[1]: Entering directory `/usr/local/openldap-2.3.38/libraries'
Making all in /usr/local/openldap-2.3.38/libraries
  Entering subdirectory liblutil
make[2]: Entering directory `/usr/local/openldap-2.3.38/libraries/liblutil'
make[2]: Nothing to be done for `all'.
make[2]: Leaving directory `/usr/local/openldap-2.3.38/libraries/liblutil'

  Entering subdirectory liblber
make[2]: Entering directory `/usr/local/openldap-2.3.38/libraries/liblber'
make[2]: Nothing to be done for `all'.
make[2]: Leaving directory `/usr/local/openldap-2.3.38/libraries/liblber'
```

图 12-15　安装 OpenLDAP 安装包

点评与拓展：安装 OpenLDAP 软件与安装其他软件有所不同，增加了建立库依赖关系和测试安装包两步。另外，第(2)步与 BerkeleyDB 4.2 建立联系很重要，这也是之所以要先安装 BerkeleyDB 4.2 的原因。

12.3.3　OpenLDAP 的启动与测试

至此，OpenLDAP 安装完毕，接下来讲述如何启动 OpenLDAP，并且对其作简单的功能测试。

❶ 首先需要配置 OpenLDAP 的域及管理用户，它的配置文件是/usr/local/openldap/etc/openladap/slapd.conf。加入如图 12-16 中画线的三行语句：第一句指定了 OpenLDAP 的根域(OpenLDAP 的域采用了这样的域名组合方式，比如说配置文件初始的 suffix　"dc=my-domain,dc=com"，说明域名是 my-domain.com，com 是其根域，my-domain 是根域之下的域名)，在图 12-16 中，我们改为" suffix　"dc=iccc"，这仅仅指定了根域为 iccc；第二句指定了根域的管理用户 admin，即 LDAP 服务器的超级用户；第三句与第二句呼应，设定了根域的管理用户 admin 的密码。以上三个参数是 OpenLDAP 的基本参数，设定好后方能执行以后的测试操作。

❷ 完成 OpenLDAP 的配置后，使用下面命令启动 LDAP 服务器，如图 12-17 所示，并且可以利用 LDAP 附带的查询命令测试 LDAP 是否成功启动，兹将两条命令分列于下：

```
/usr/local/openldap/libexec/slapd
/usr/local/openldap/ldapsearch -x -b '' -s base '(objectclass=*)'
 namingContexts
```

如图 12-17 所示，输入查询命令后，可以看到查到了 dc=iccc 的记录，记录条数为 1 条，说明 LDAP 服务器已经成功启动。也可以输入下面命令查看 slapd 进程是否启动，图 12-18 显示了查询结果，画线部分就是 LDAP 服务器的守护进程，进程号为 5865。

```
ps -aux | grep slapd
```

ps 是查看系统进程命令，-aux 是需要列出的进程参数，| grep slapd 表示只列出进程启动程序为 slapd 的那些进程号。

```
#######################################################################
# BDB database definitions
#######################################################################

database        bdb
#suffix         "dc=my-domain,dc=com"
suffix "dc=iccc"
rootdn "cn=admin,dc=iccc"
#rootdn         "cn=Manager,dc=my-domain,dc=com"
# Cleartext passwords, especially for the rootdn, should
# be avoid.  See slappasswd(8) and slapd.conf(5) for details.
# Use of strong authentication encouraged.
rootpw          iccc
# The database directory MUST exist prior to running slapd AND
# should only be accessible by the slapd and slap tools.
# Mode 700 recommended.
directory       /usr/local/openldap/var/openldap-data
# Indices to maintain
index   objectClass     eq
```

图 12-16　配置 slapd.conf

```
[root@seugrid3 openldap]# libexec/slapd
[root@seugrid3 openldap]# ldapsearch -x -b '' -s base '(objectclass=*)' namingContexts
# extended LDIF
#
# LDAPv3
# base <> with scope baseObject
# filter: (objectclass=*)
# requesting: namingContexts
#

#
dn:
namingContexts: dc=iccc

# search result
search: 2
result: 0 Success

# numResponses: 2
# numEntries: 1
[root@seugrid3 openldap]#
```

图 12-17　启动 OpenLDAP

```
[root@seugrid3 openldap]# ps -aux | grep slapd
Warning: bad syntax, perhaps a bogus '-'? See /usr/share/doc/procps-3.2.6/FAQ
root      5865  0.0  0.2  17040  2632 ?        Ssl  19:47   0:00 libexec/slapd
root      5888  0.0  0.0   5156   732 pts/2    S+   19:48   0:00 grep slapd
[root@seugrid3 openldap]#
```

图 12-18　查看 LDAP 进程

❸ 可以使用下面命令安全关闭 LDAP 服务器，当然也可以通过使用 ps 命令查出 slapd 的进程号，然后使用 kill 命令关闭该进程的方法关闭 LDAP 服务器。

```
kill -INT 'cat /usr/local/var/slapd.pid'
kill <pid>
```

如图 12-18 所示的 LDAP 服务器的进程号为 5865。应注意画线下面一行是刚刚运行的 ps 命令的进程，不是 LDAP 服务器的进程，因此使用下面命令就可以关闭 LDAP 服务器。

```
kill 5865
```

❹ 当需要写入 LDAP 服务器数据时，我们通常不通过 ladapadd 命令逐条输入，而是将需要写入的信息存入后缀名为.ldif 的文件中，通过读取整个文件完成 LDAP 数据库的输入。ldif 代表了 LDAP 数据交换格式，也可以将 ldif 文件看作供人阅读的 LDAP 信息，它是提供和传送 LDAP 数据的标准文件。

我们创建一个简单的名字为 first.ldif 的文件来讲述 ldif 文件的格式及 LDAP 服务器是如何读取 ldi 文件的。使用 vi 命令打开 first.ldif 文件如图 12-19 所示，输入图示文本内容，ldif 文件由如下格式的文本组成：

```
dn: dc=<MY-DOMAIN>,dc=<COM>
objectclass: dcObject
objectclass: organization
o: <MY ORGANIZATION>
dc: <MY-DOMAIN>

dn: cn=Manager,dc=<MY-DOMAIN>,dc=<COM>
objectclass: organizationalRole
cn: Manager
```

这里当然需要将<MY-DOMAIN>和 cn=Manager 替换成步骤(1)slapd.conf 配置文件中已填好的内容。需要特别注意的是，文件中每一行字符结束后，不能留有空格字符，当在 Windows 下编辑该文本再拷入 Linux 时常会出现这种错误，这样将导致 LDAP 服务器无法识别该 ldif 文件。

```
[root@seugrid3 openldap]# vi first.ldif
dn: dc=iccc
objectclass: dcObject
objectclass: organization
o:iccc
dc: iccc

dn: cn=admin,dc=iccc
objectclass: organizationalRole
cn: admin
```

图 12-19　编辑 first.ldif 文件

❺ 编辑完 first.ldif 文件后，将其拷入 openldap 的安装目录，就可以使用下面 ldapadd 命令加载该文件，如图 12-20 所示，shell 提示输入密码，只要输入前面 slapd.conf 配置文件中所设定的密码即可。

```
/usr/local/openldap/bin/ldapadd -x -b "cn=admin,dc=iccc" -W -f first.ldif
```

如果 first.ldif 文件书写无误，shell 提示添加了两条记录，如图 12-20 所示；如果 first.ldif 文件有误，系统将提示读到的第一条错误，具体错误原因可以查询 FAQ 网页：http://www.openldap.org/faq/data/cache/53.html。

```
[root@seugrid3 openldap]# bin/ldapadd -x -D "cn=admin,dc=iccc" -W -f first.ldif
Enter LDAP Password:
adding new entry "dc=iccc"

adding new entry "cn=admin,dc=iccc"

[root@seugrid3 openldap]#
```

图 12-20　使用 ldapadd 加载 first.ldif

❻ 加载 ldif 文件成功后，可以输入下面命令查询添加后的 LDAP 记录。

```
/usr/local/openldap/ldapsearch -x -b 'dc=iccc' '(objectclass=*)'
```

查询命令的意思是，查询 LDAP 中所有域名为 iccc 的记录项，图 12-21 显示了查询结果，一共有两项记录，正是上面在 ldif 文件中书写的。

```
[root@seugrid3 openldap]# ldapsearch -x -b 'dc=iccc' '(objectclass=*)'
# extended LDIF
#
# LDAPv3
# base <dc=iccc> with scope subtree
# filter: (objectclass=*)
# requesting: ALL
#

# iccc
dn: dc=iccc
objectClass: dcObject
objectClass: organization
o: iccc
dc: iccc

# admin, iccc
dn: cn=admin,dc=iccc
objectClass: organizationalRole
cn: admin

# search result
search: 2
result: 0 Success

# numResponses: 3
# numEntries: 2
[root@seugrid3 openldap]#
```

图 12-21　查询 LDAP 记录

❼ OpenLDAP 除了前面提及的 ldapadd 和 ldapsearch 命令，还提供了其他 LDAP 命令行工具，

在此一一列出，而不再逐一演示。

ldapdelete：删除 LDAP 服务器中存在的记录。

ldapmodify：添加、修改和删除 LDAP 服务器中存在的记录。

ldapmodrdn：修改 DN 的最左端部分相对与众不同的 LDAP 服务器中的记录。

点评与拓展：我们在 LDAP 概述部分讲过 LDAP 层状的目录结构，在本节的安装过程中，我们看到安装起始是创建了 LDAP 层状目录结构的根域，用户则在根域下面扩展所需要的子域。

LDAP 的层状目录结构决定了其查询效率高的特点，因此，LDAP 被广泛应用于海量数据的分布式检索领域。在 LDAP 安装并添加入数据后，下一步的问题就是 LDAP 编程，即如何通过其他应用程序调用 LDAP。复杂的编程都源自 5 个主要 LDAP 操作：连接、绑定、执行 LDAP 活动、解除绑定和断开连接。

由于本书论述范围及篇幅所限，对 LDAP 的论述就到安装、载入 LDAP 数据为止，至于如何开发 LDAP 应用程序，请读者参考 Wrox 的 Implementing LDAP 一书，相信此书一定能开启您在 LDAP 方面的千里之行！

12.4　架设 LDAP 服务器及管理平台

应用实例导航——LDAP 服务器的管理

※场景呈现

A 公司在安装好 LDAP 服务器后，需要在根域 iccc 下为各个分公司建立独立子目录，用来存储员工的信息。为了方便地管理 LDAP 服务器，A 公司希望使用图形化管理平台来方便地对 LDAP 服务器的记录进行添加、修改和删除。

LDAP 目录的层次结构要求为：根域的名字为 iccc，下面建立两个子域，名字分别为 company_andy 和 company_bob，另外建立一个根域 iccc 下的组织，名字为 company_Jonghu，并在该组织内示例性地为一个名字为 Jeshope 的员工建立记录。

※技术要领

(1) 掌握 LDAP Browser/Editor 的安装步骤。

(2) 掌握 LDAP Browser/Editor 的使用方法。

(3) 学会从浏览器查看 LDAP 服务器。

尽管使用 shell 命令载入.ldif 的方法也可以执行添加、修改和删除 LDAP 服务器记录的相关操作，但是这种方法对用户要求较高，不直观。于是，用户就希望以图形化界面的方法来高效快捷地管理 LDAP 服务器，这需要借助相关 LDAP 浏览编辑软件来实现。LDAP

Browser/Editor 就是实现类似功能的合适软件，它提供用户界面友好的 LDAP 目录编辑工具，由 JFC(SwingSet)和 JNDI 类库写成，可以连接第二版和第三版 LDAP 服务器，并可以免费下载使用。

本节将利用 LDAP Browser/Editor 软件来实现上面应用实例的要求，分两节讲述：12.4.1 节讲述 LDAP Browser/Editor 软件的下载和安装；12.4.2 节讲述如何使用 LDAP Browser/Editor 来实现对 LDAP 服务器的相关操作。

12.4.1 LDAP Browser/Editor 的下载和安装

本节讲述 LDAP Browser/Editor 软件的下载和安装，步骤如下。

❶ 网站http://www-UNIX.mcs.anl.gov/~gawor/ldap/download.html提供了 LDAP Browser/Editor 软件的免费下载，如图 12-22 所示。本书附带光盘也提供了这一软件。

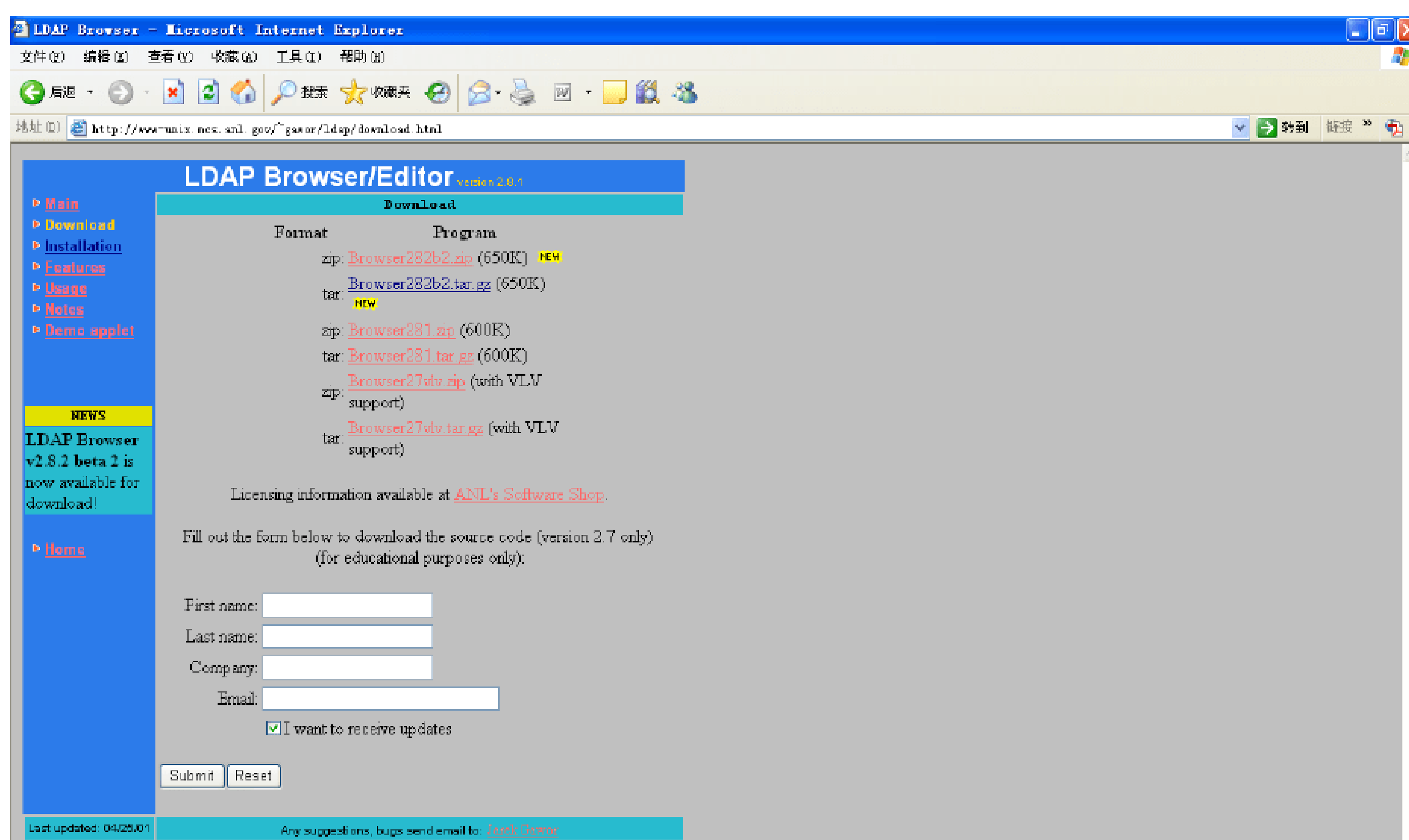


图 12-22 LDAP Browser/Editor 软件的下载

❷ 下载的安装包名字为 Browser282b2.tar.tar，LDAP Browser/Editor 同时适用于 Windows 和 Linux 系统，本书详细讲述如何在 Linux 系统下使用它。将安装包复制到/usr/local 目录，利用 tar 命令解压缩，形成/usr/local/ldapbrowser 目录，进入该目录可以看到如图 12-23 所示的文件，其中 lbe.bat 是 Windows 下的脚本文件，如果在 Windows 下运行 LDAP Browser/Editor，就需要执行该脚本，该脚本读取了系统的 JAVA_HOME 环境变量，系统必须首先正确设定 JAVA_HOME 后，该脚本方能运行成功。在 Linux 系统下运行 LDAP Browser/Editor 时，需要以 X-Window 图形化方式登录，lbe.sh 读取系统的 JAVA_HOME 环境变量，如图 12-24 所示，同样要在 JAVA_HOME 环境变量正确设定后，方可运行 lbe.sh 入口程序。

```
[root@seugrid3 ldapbrowser]# ls
applet                    help        lbe.sh          relnotes.html
attributes.config         lbe.bat     LdapBrowser.lnk templates
attributes.config.sample  lbe.jar     lib             uofmichigan.cfg
CHANGES.TXT               lbe.old.bat LICENSE.ICONS   uofmichigan.cfg.sample
faq.html                  lbe.old.sh  readme.html
[root@seugrid3 ldapbrowser]# ./lbe.sh
```

图 12-23　解压后的 LDAP Browser/Editor 目录

```
[root@seugrid3 ldapbrowser]# vi lbe.sh
#!/bin/sh

OPTIONS=

if [ ! -d "$JAVA_HOME" ] ; then
    JAVA=java
  else
    JAVA=${JAVA_HOME}/bin/java
fi

if [ "$1" = "fix13" ] ; then
    OPTIONS="${OPTIONS} -Xbootclasspath/p:lib/ldap.jar:lib/jndi.jar:lib/providerutil.jar:lib/ldapbp.jar"
    ARG1=
  else
    ARG1=$1
fi

${JAVA} ${OPTIONS} -jar lbe.jar $ARG1 $2 $3 $4 $5 $6 $7 $8 $9
```

图 12-24　lbe.sh 脚本文件内容

12.4.2　LDAP Browser/Editor 的使用

本节介绍如何使用 LDAP Browser/Editor 软件连接 LDAP 服务器，并对 LDAP 目录进行操作，具体步骤如下。

❶ 启动 LDAP Browser/Editor 后，弹出如图 12-25 所示的 Connect 窗口，用于设置连接的相关属性。首先对 Quick Connect→Connection 选项卡进行设定，需要在 Host 文本框中输入 LDAP 服务器的 IP 地址，本例中是 172.18.12.179； Port 文本框中输入 LDAP 服务器的端口，其默认端口是 389； Version 文本框中输入 LDAP 的版本，默认值是 3。然后，单击 Fetch DNs 按钮，在 Base DN 文本框中就出现根域的名称：dc=iccc，标识了所要登录的 LDAP 服务器的根域；接着，取消选中 SSL 和 Anonymous bind 两个复选框，并选中 append base DN 复选框，在 User DN 文本框中输入 slapd.conf 配置文件中所设置的管理用户名字，本例中是 cn=admin。

❷ 其次，需要对 Quick Connect→Options 选项卡进行设定，设定界面如图 12-26 所示。在 Referrals 选项选择 Manage 复选框，因为下面要对 LDAP 服务器记录进行修改。其他选项可以根据用户喜好选择。

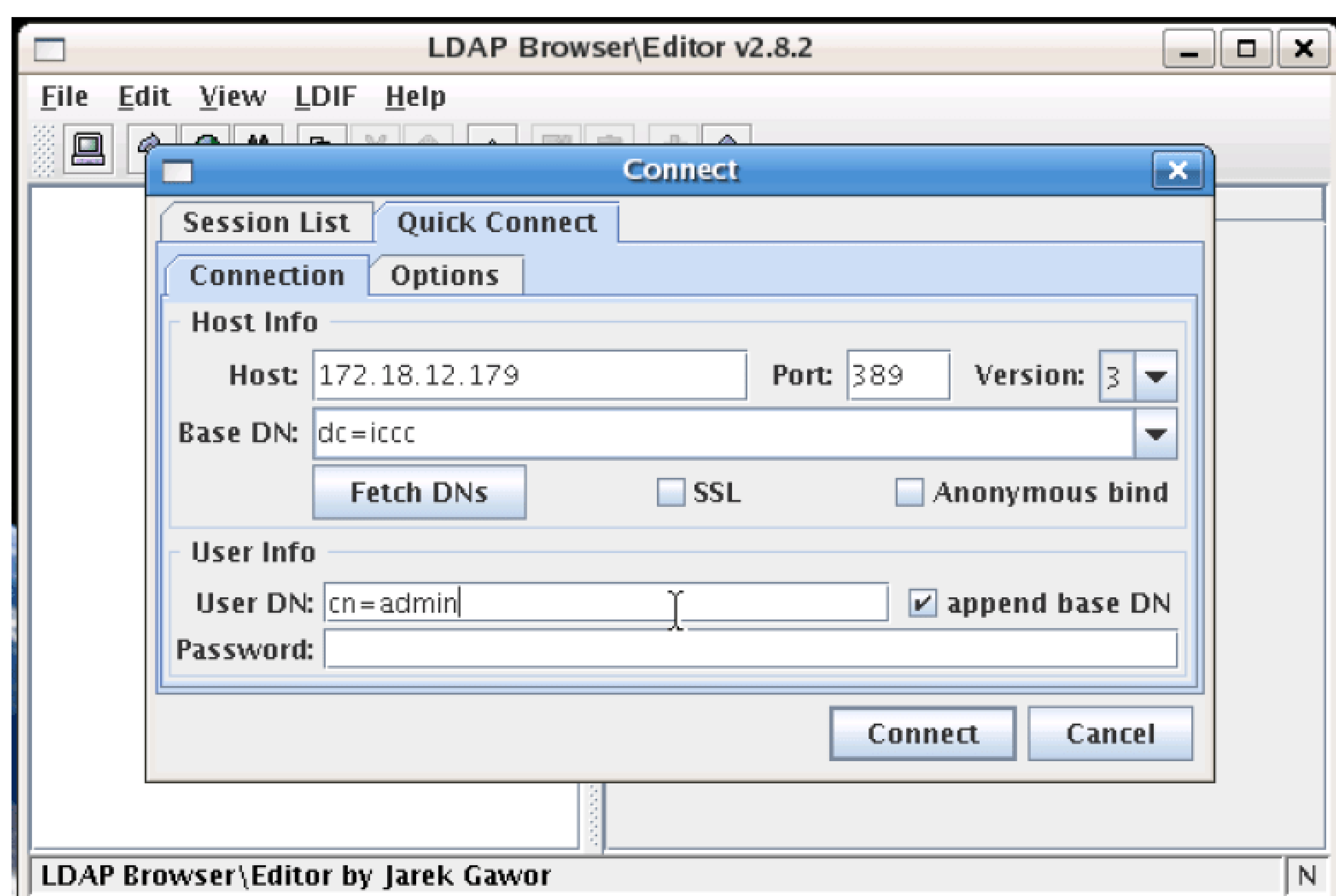


图 12-25 LDAP Browser/Editor 的 Connection 窗口

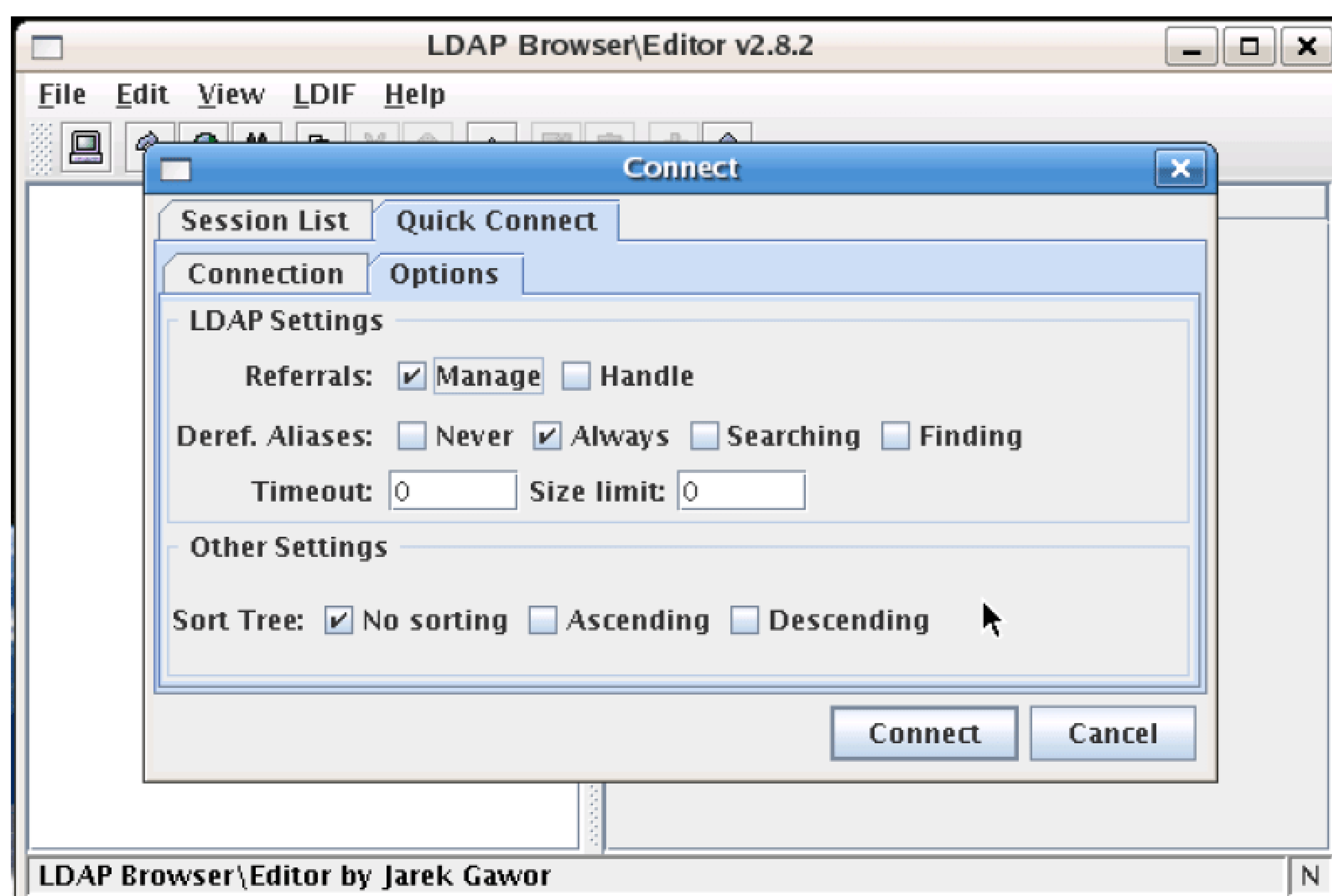


图 12-26 LDAP Browser/Editor 的 Options 配置

❸ 对 Quick Connect 选项卡设定完毕后，单击 Connect 按钮连接 LDAP 服务器，弹出 Enter Password 对话框如图 12-27 所示，要求输入密码，正确输入后，即可登录。登录成功后，出现如图 12-28 所示的 LDAP 服务器的初始界面，显示的记录正是从前面 first.ldif 文件中载入的内容。

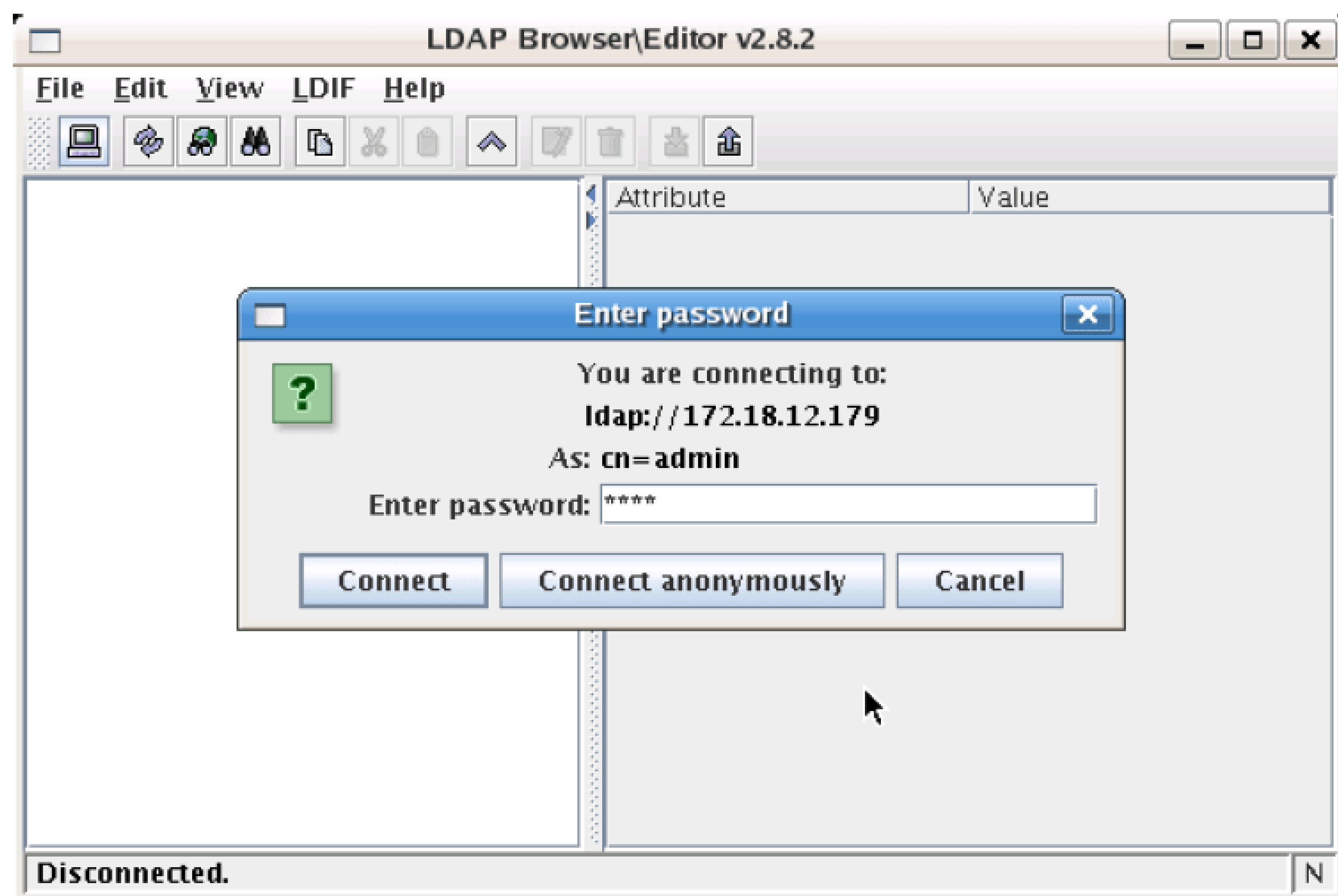


图 12-27　连接 LDAP 服务器输入密码

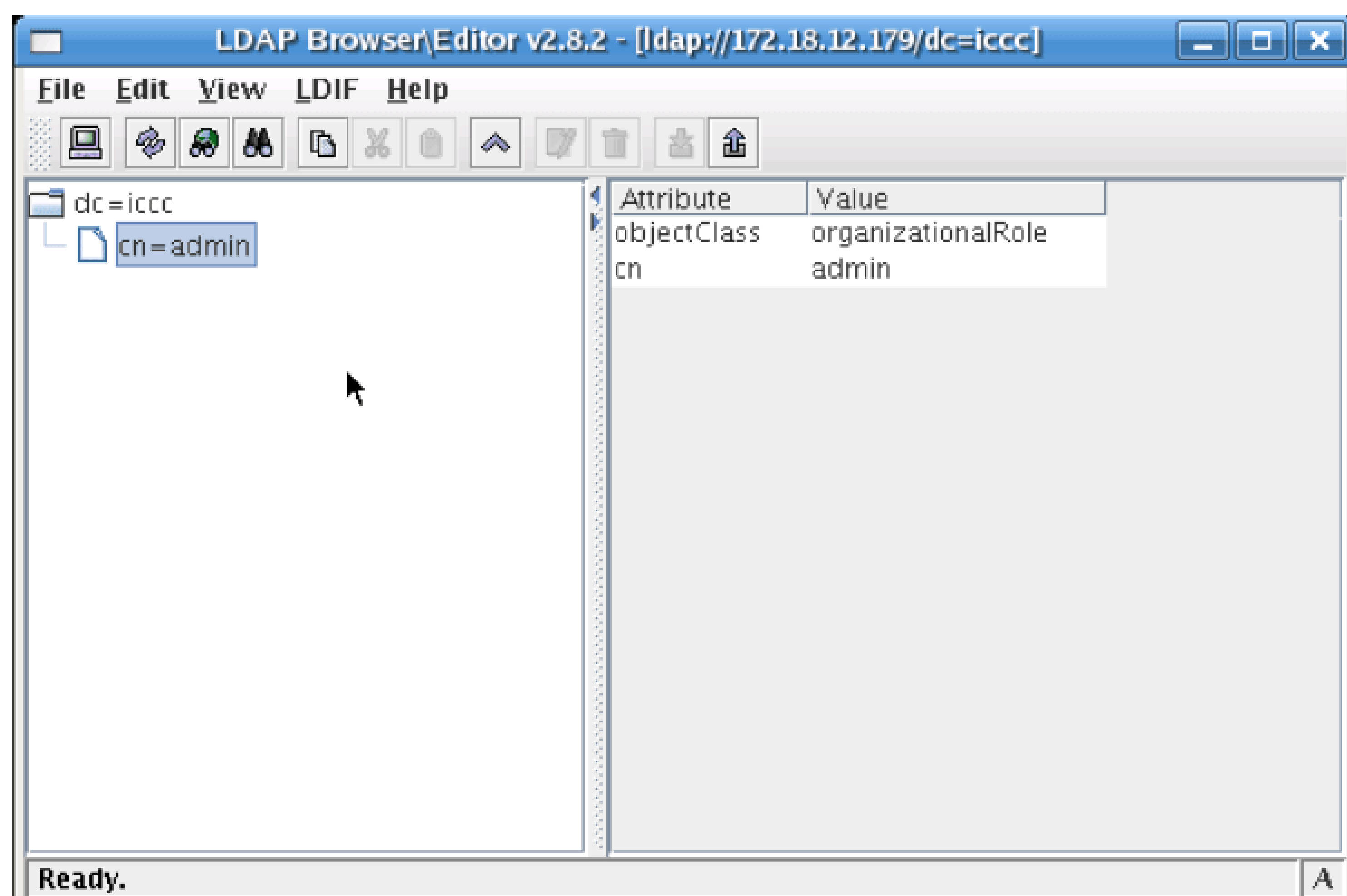


图 12-28　LDAP 服务器的初始界面

❹ 接下来，我们演示如何创建实例场景所要求的名字为 company_Jonghu 的组织。选中 dc=iccc，选择 Edit→Add Entry→organization 命令，如图 12-29 所示，弹出如图 12-30 所示的填写组织基本信息界面。用户可以根据具体需要填写，本例中仅代表性地填写了几项，填完之后，单击 Apply 按钮使之生效。另外，其中的 userPassword 为设置密码选项，如果直接输入密码，则为明文显示；如果单击 set 按钮输入，则为密文显示。单击 Verify 按钮可以验证密码。

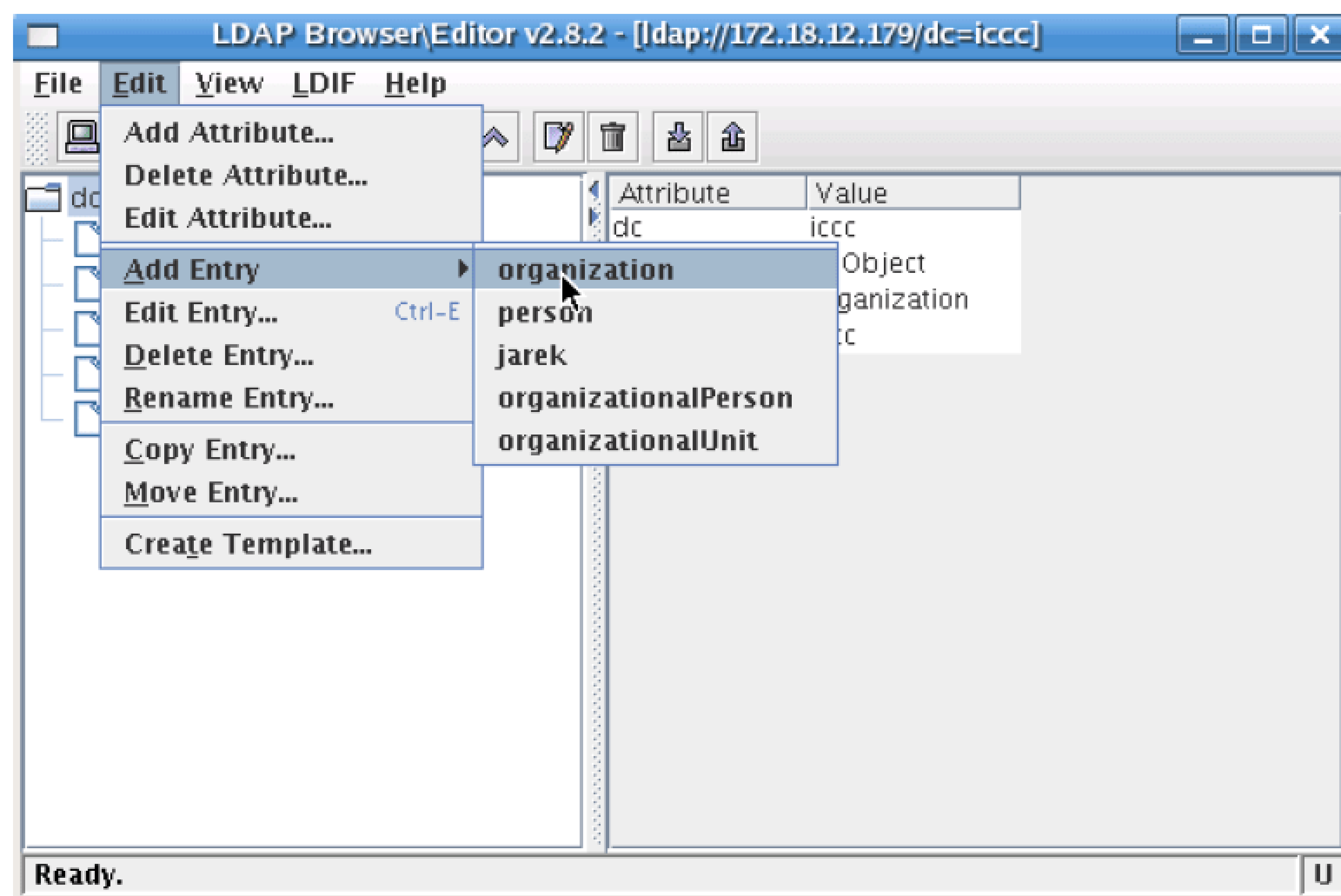


图 12-29　创建组织

Create New 'organization' Entry

File Edit

dn:	o=company_Jonghu, dc=iccc
objectclass:	top
objectclass:	organization
businessCategory:	JiangSu Nanjing
postOfficeBox:	
streetAddress:	
postalCode:	210096
searchGuide:	
facsimileTelephoneNumber:	
userPassword:	Verify Set Save as Insert from
preferredDeliveryMethod:	
telephoneNumber:	
physicalDeliveryOfficeName:	
registeredAddress:	
destinationIndicator:	
st:	
x121Address:	
l:	
postalAddress:	
seeAlso:	
description:	
telexNumber:	
internationaliSDNNumber:	
teletexTerminalIdentifier:	

Apply Cancel

图 12-30　填写创建组织基本信息界面

❺ 添加组织成功后，可以看到根域 dc=iccc 下出现了刚才的记录：o=company_Jonghu，如图 12-31 所示，标注部分是我们上一步所填写的基本信息。

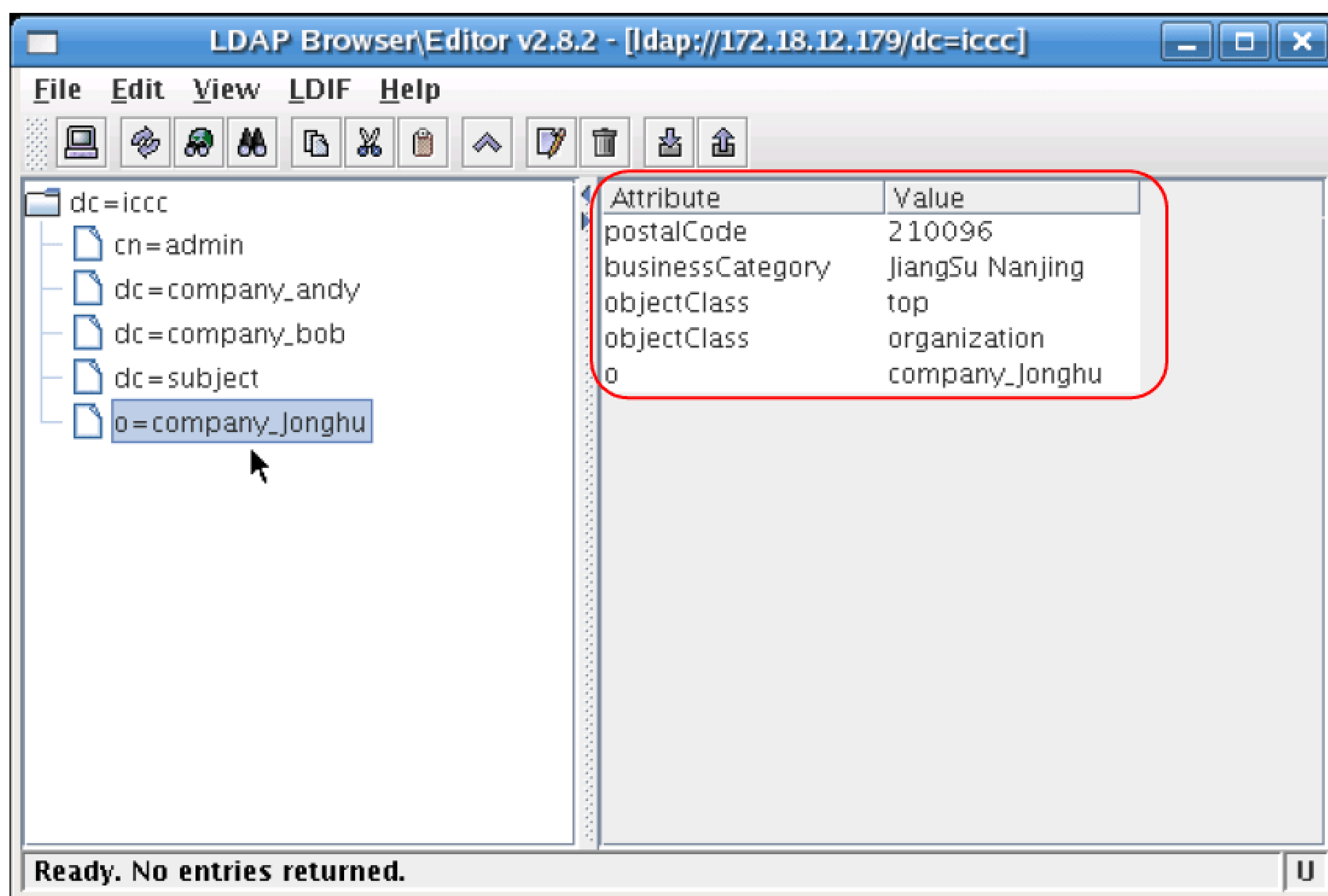


图 12-31　加入 company_Jonghu 后的结果显示

❻ 接着我们在 company_Jonghu 的组织中创建实例场景所要求的名字为 Jeshope 的员工。选中 o=company_Jonghu 记录，选择 Edit→Add Entry→person 命令，如图 12-32 所示。

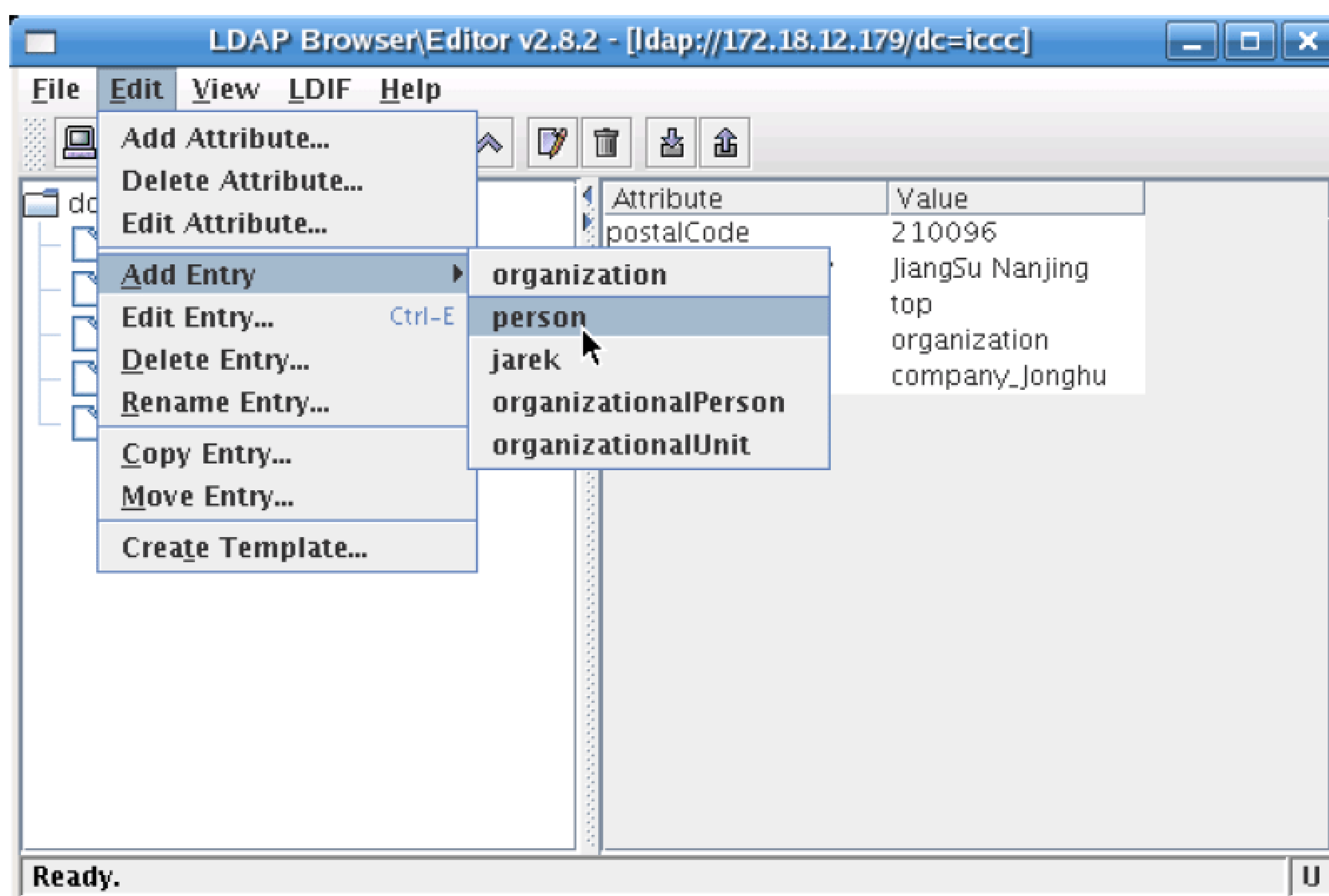


图 12-32　创建个人信息

❼ 随后出现的图 12-33 显示了填写个人基本信息的界面，将 dn 文本框中的 cn 参数改为 Jeshope，最后一项 sn 文本框中的内容要与 cn 一致，这里也必须为 Jeshope。

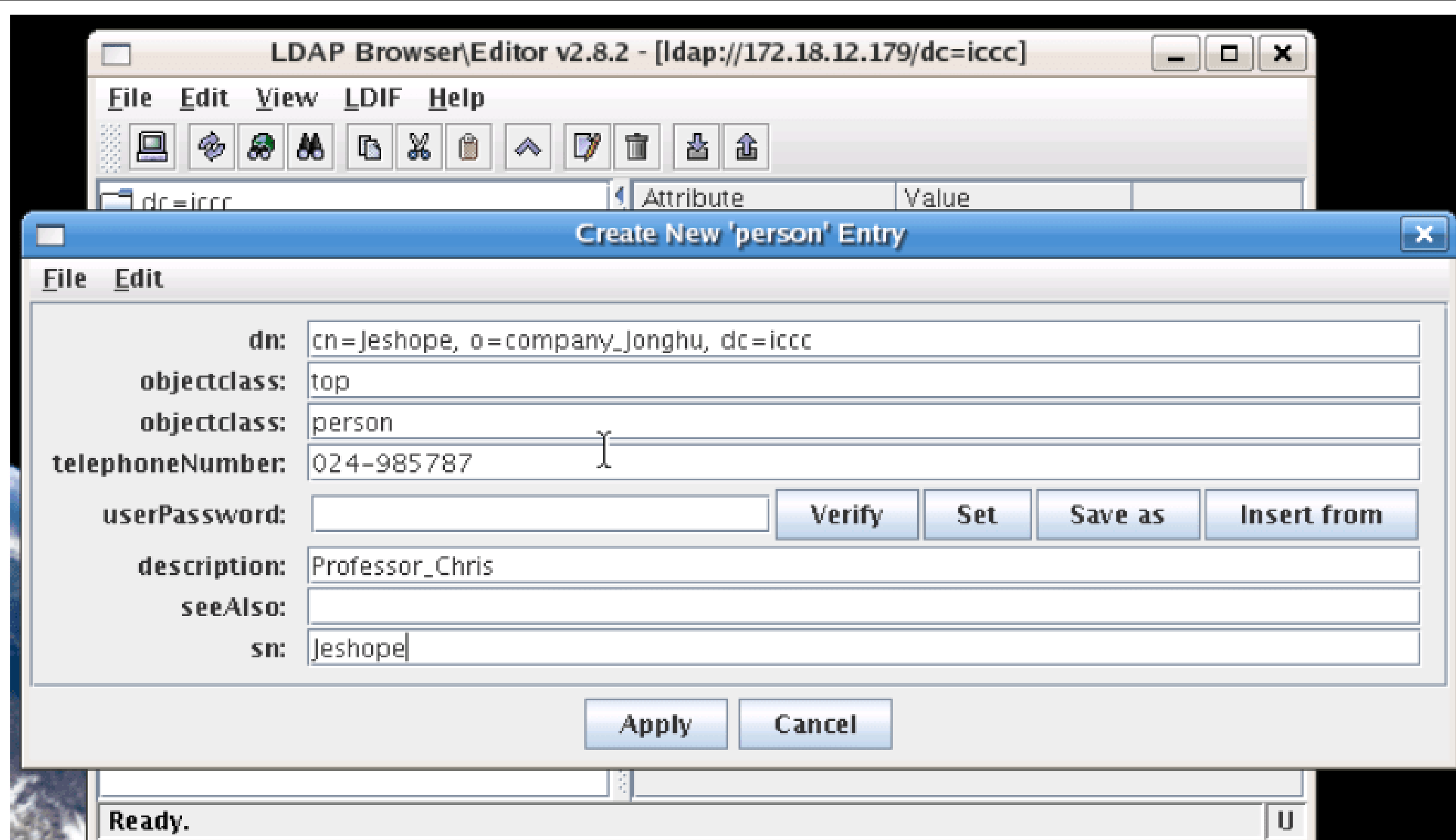


图 12-33　填写创建个人的基本信息

❽ 单击 Apply 按钮，可以看到组织 o=company_Jonghu 的记录下，出现了 cn=Jeshope 的记录，如图 12-34 所示，右边子窗口为上一步所填写的基本信息。

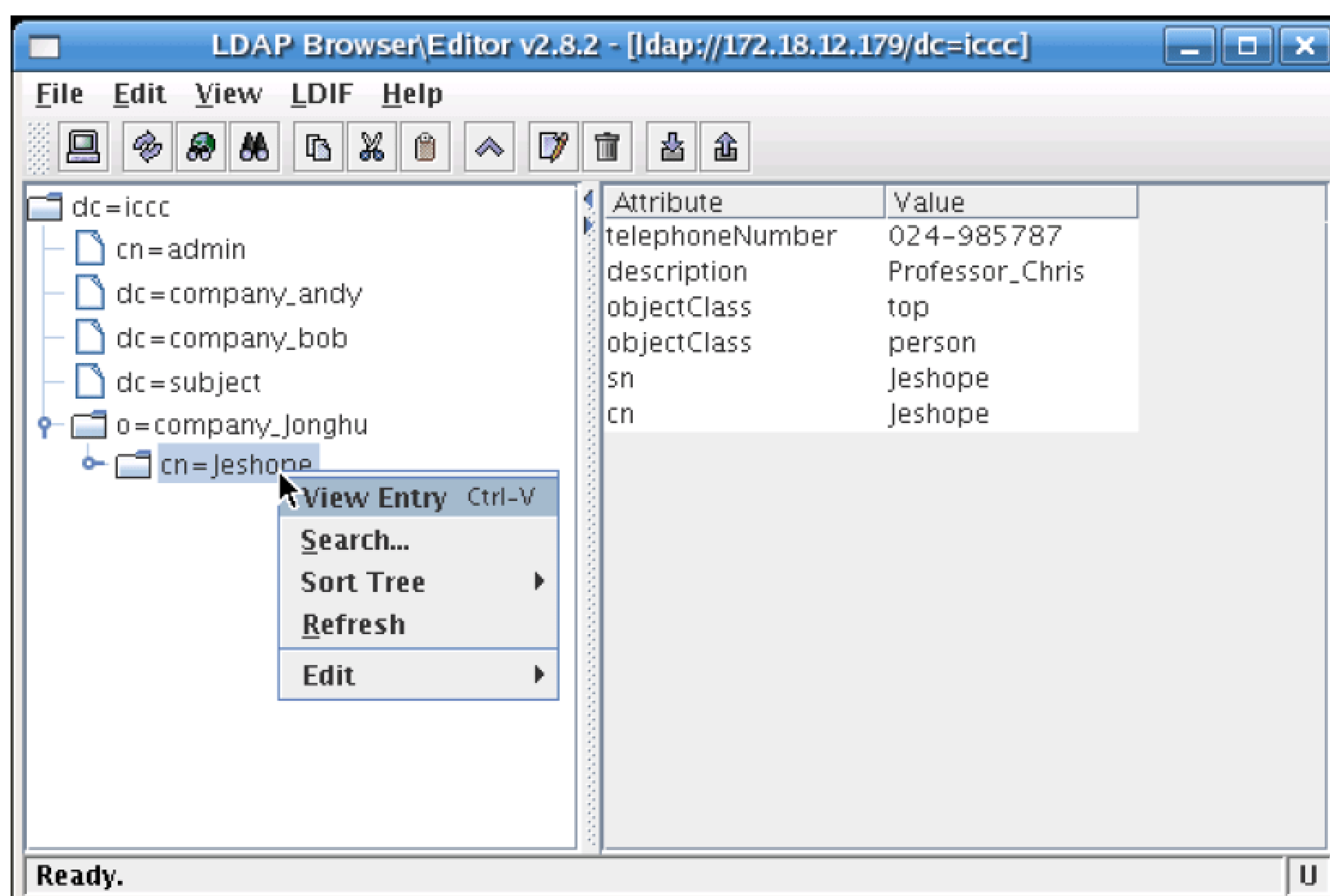


图 12-34　Jeshope 个人信息显示

❾ 至此，已经介绍了如何在服务器端架设 LDAP 服务器。为了能方便地管理和架设 LDAP 服务器，我们借助 LDAP Browser/Editor 软件以图形化的方式配置 LDAP 服务器。为使 LDAP 服务器得到更广泛的应用，其数据必须可以从网络上访问到，即可以通过 Web 浏览器查看 LDAP 数据的记录，为此，IETF(因特网工程任务组)制定了 LDAP 的 URL 标准，该 URL 标准与常规的 HTTP URL 相似。RFC2255 中规定了 LDAP URL 的语法，如下所示：

```
ldap://<hostname>/<base>?<attrs>?<scope>?<filter>
```

其中参数解释如下。

ldap://：指定 Web 浏览器所访问的协议类型；

hostname：指定 LDAP 服务器的主机名或 IP 地址；

base：指定搜索参数；

attrs：指定属性列表，用逗号分开；

scope：指定搜索范围(sub、one 或 base)；

filter：指定搜索过滤器。

比如，我们需要搜索名字为 Jeshope 的记录，输入 ldap://172.18.12.179/cn=Jeshope，o =company_Jonghu,dc=iccc 就可以访问。Microsoft 公司的 Internet Explorer 以地址簿格式显示搜索结果，如图 12-35 所示。Netscape Communicator 浏览器能够以网页的形式显示 LDAP 数据。

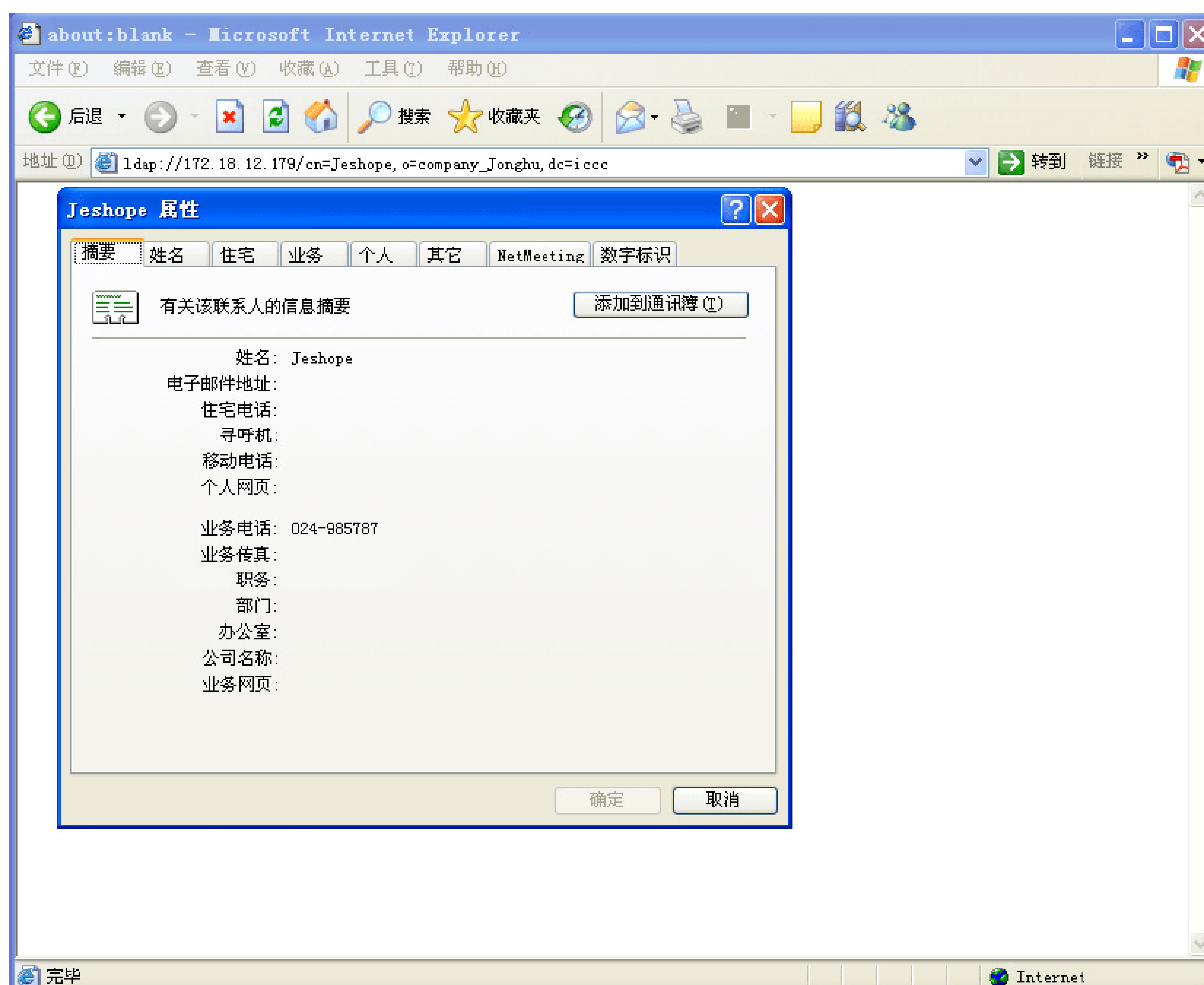


图 12-35　访问 LDAP 服务器

点评与拓展： 在架设完一个简单的示例性 LDAP 服务器后，相信读者已经对安装、架设 LDAP 服务器的步骤有所了解。对于复杂一些的 LDAP 服务器的架设，基本的步骤与本例差不多，只要稍加修改和扩展即可。

12.5 本章小结

本章介绍了目录服务的基本概念，以及轻量目录访问协议(LDAP)的层次式存储树原理。主要介绍了如何利用 OpenLDAP 软件架设 LDAP 服务器，其中包括 Berkerley 数据库的安装、OpenLDAP 的安装、如何载入.ldif 数据文件以及 LDAP 的基本命令等内容。最后通过一个实例讲述了如何利用 LDAP Browser/Editor 软件图形化的管理用 OpenLDAP 软件所架设的 LDAP 服务器，具体内容包括 LDAP Browser/Editor 软件的安装和配置、如何使用 LDAP Browser/Editor 软件连接 LDAP 服务器，并构建层次式目录树。

第 13 章　Samba 服务器的配置与架设

在大多数局域网内，都是 Linux、Windows XP、Windows Server、UNIX 等多操作系统并存的情形，如何实现多操作系统之间的资源共享、网络互联成为了架设局域网需要考虑的问题之一。

架设 Samba 服务器就是实现多系统间互联、共享的一个很好的解决方案，它使 Windows 客户端能通过“网上邻居”所提供的工作组功能访问 Linux 服务器上的资源，也使 Linux 客户端能通过服务管理块协议方便地访问 Windows 服务器的资源。本章将介绍 Samba 服务的概念和原理、Samba 服务器端的配置和启动以及 Windows 系统和 Linux 系统 Samba 客户端的配置方法。

通过本章的学习，读者应掌握以下内容：

- ✧ Samba 服务的概念和意义
- ✧ Samba 服务器端的配置和启动
- ✧ Samba 客户端的配置和测试

13.1　Samba 服务概述

13.1.1　Samba 简介

在介绍 Samba 软件之前，先来介绍一下 SMB(Server Message Block，服务信息块)协议。SMB 协议是局域网上共享文件夹/打印机的一种协议。该协议可以用在 TCP/IP 之上，也可以用在其他网络协议之上。客户端程序借助 SMB 协议可以在各种网络环境下读写服务器上的文件，对服务器程序提出请求，还可以访问远程服务器端的文件或打印机资源等。SMB 可以用于包括 Linux 的多种平台。

Linux 使用 Samba 程序来实现 SMB 协议。安装 Samba 后，Linux 系统就变成了一台 SMB 服务器，而 Windows 用户也能够使用 Linux 系统下的共享文件和打印机等资源。Linux 用户同样也能够利用 SMB 服务器来访问 Windows 下的共享文件和打印机等资源。

Samba 属于 GPL(GNU public license)软件，任何人都可以免费使用，用户可以从 Samba 公司的官方网站上下载。Samba 的主要功能有：

- ✧ 可以在 Windows 网络中解析 NetBIOS 名字。网络上各主机都要定期向网上广播各自的身份信息，一方面是为了利用网上资源，另一方面也是为了让别人使用自己的资源。负责收集这些信息并提供检索的服务器称为浏览服务器。而 Samba 就可

以充当这样一种浏览服务器的角色，并且在跨网关的时候还可以充当 WINS 服务器。

- ✧ 提供了 Windows 风格的文件和打印机共享。Windows 95 以上的 Windows 操作系统都可以利用 Samba 访问 Linux 等其他操作系统下的共享资源，而外表风格上则与访问 Windows 下共享资源的方法没什么区别。
- ✧ 提供了一个命令行工具，从而可以有限制地支持 Windows 的某些管理功能。
- ✧ SMB 客户功能。Samba 提供的 smbclient 程序可以让用户在 Linux 上以类似 FTP 的方式访问 Windows 的一些共享资源。

13.1.2 Samba 服务工作原理

Samba 服务让 SMB 和 NetBIOS 协议运行在 TCP/IP 协议上，利用 NetBEUI 协议使 Windows 用户可以在“网上邻居”中看到 Linux 系统中的资源，同时也让 Linux 客户端可以访问 Samba 服务器上的资源。图 13-1 描述了 Samba 服务的工作流程，具体步骤如下。

Samba 客户端首先发送一个协商请求数据包给 Samba 服务器端，列出它所支持的 SMB 协议。服务器收到请求后作出响应，根据客户端所支持的协议版本选择所希望使用的协议版本。

协议版本确定后，客户端发送一个建立会话请求给服务器，服务器作出响应，同意或拒绝建立本次连接。

完成协商及认证后，客户端会发送一个请求数据包给服务器，其中包含了它欲访问的网络资源名称。服务器收到请求后，就会发送一个响应信息给客户端，表示接受或拒绝此次连接。建立连接后，SMB 客户端就可以使用 open SMB 打开一个文件，使用 read SMB 读取文件，使用 write SMB 写入文件，使用 close SMB 关闭文件。



图 13-1 Samba 服务的工作流程

13.2　Samba 服务器端的配置

应用实例导航——A 公司架设 Samba 服务器

※场景呈现

A 公司有一台 Linux 服务器，地址为 172.18.12.179；大部分用户使用的是 Windows XP 的 PC 机，IP 范围在 172.18.12.*内。现在需要利用 Samba 服务将 Linux 服务器上的/home/sharing 目录开放给所有主机(包括 Linux、Windows XP)共享，要求 Samba 服务的用户是 tian，并开放 tian 的根目录；但是，要求/home/sharing 开放可写权限，tian 根目录仅开放可读权限。

另外，要求 Windows XP 上的用户 winsamba，具有访问 Linux 服务器/home/sharing 目录的权限。

※技术要领

(1) Samba 服务器端的主配置。
(2) Samba 服务用户的设定。
(3) Samba 服务与 Windows 用户的映射。
(4) Samba 服务的启动。

本节介绍 Samba 服务器端的配置方法，重点介绍 Samba 服务主配置文件/etc/samba/smb.conf 的意义和设定方法；然后介绍如何设定 Samba 服务器的用户，以及如何与 Windows 用户建立映射关系；最后介绍 Samba 服务的启动。

13.2.1　Samba 服务器端设定

Samba 服务的主配置文件为/etc/samba/smb.conf，包括了该 Samba 服务器的全局参数、信任主机和网域、共享目录、安全设置等内容。/etc/samba/smb.conf 中有两种注释符号：分号“;” 和井号“#”，以这两种符号开头的行是注释行，即其内容被忽略而不会生效。该配置文件以“设定项=设定值”的格式来表示。下面介绍/etc/samba/smb.conf 的意义和设定方法。

❶ 首先，输入下面命令确认 Samba 服务是否安装，如图 13-2 所示，可以看到 Linux 系统的 Samba 版本为 3.0.21，图 13-2 中标注的组件即为 Samba 服务器端的组件。

```
rpm -qa | grep samba
```

```
[root@sec ~]# rpm -qa | grep samba
samba-client-3.0.21b-2
system-config-samba-1.2.34-1
samba-common-3.0.21b-2
samba-3.0.21b-2
```

图 13-2　检查 Samba 是否安装

❷ 打开/etc/samba/smb.conf 文件，首先进行全局设定，这些设定都是与 Samba 服务整体运行参数有关的设定项，对所有的共享目录都生效。找到 smb.conf 文件中的[global]标签，该标签的以下部分都为全局设定部分。第一处是设定工作组名，如图 13-3 第一处标注所示，这个工作组名不是随意设的，必须要与 Windows 下的工作组名一样。那么怎么知道 Windows 的工作组名呢？我们可以通过以下方法获得：在桌面上右击【我的电脑】图标，在弹出的快捷菜单中选择【属性】命令，然后在【系统属性】对话框中切换到【计算机名】选项卡，可以看到计算机名窗口中有一行显示了本机所属的工作组，如图 13-4 所示，本例的工作组名为 MSHOME。于是，将 workgroup 设为 MSHOME。

```
#======================= Global Settings =====================================
[global]

# workgroup = NT-Domain-Name or Workgroup-Name
   workgroup = MSHOME

# server string is the equivalent of the NT Description field
   server string = MSHOME Samba Server

# This option is important for security. It allows you to restrict
# connections to machines which are on your local network. The
# following example restricts access to two C class networks and
# the "loopback" interface. For more examples of the syntax see
# the smb.conf man page
   hosts allow = 172.18.12.

# if you want to automatically load your printer list rather
# than setting them up individually then you'll need this
   printcap name = /etc/printcap
   load printers = yes

# It should not be necessary to spell out the print system type unless
# yours is non-standard. Currently supported print systems include:
# bsd, sysv, plp, lprng, aix, hpux, qnx
;   printing = cups

# This option tells cups that the data has already been rasterized
cups options = raw

# Uncomment this if you want a guest account, you must add this to /etc/passwd
# otherwise the user "nobody" is used
;  guest account = pcguest
```

图 13-3　全局设定

在 workgroup 下面，可以看到 server string 设定项，这是设定 Samba 服务器的名称，将它设为 MSHOME Samba Server，如图 13-3 第二处标注所示。

第三处是设定信任主机，即允许访问所架设的 Samba 服务器的主机 IP 或网段，本例对局域网 172.18.12.*内的主机开放，如图 13-3 第三处标注所示。这里可以设定多个信任网域，比如我们还要对 192.168.10.*的子网开放，那么只需作如下设定：

```
hosts_allow=172.18.12. , 192.168.10.
```

注意，不完全 IP 后面的 “.” 分隔符不能缺少，不同于网域 IP 间用逗号隔开。

如果需要设置访问 Samba 服务器的 guest 用户，可以在如图 13-3 第四处标注部分设定，默认为不设置。

在全局设定部分，还需要设置 Samba 用户的密码文件所在位置。找到 encrypt passwords 这一行，默认是注释行，将注释符取消，表示登录 Samba 服务器需要验证密码，并且以加密的方式发送密码到 Samba 服务器端；然后将下一行的 smb passwd file 注释符也取消，表示 Samba 服务器的密码文件为/etc/samba/smbpasswd，如图 13-5 所示。这个文件是 smbpasswd 命令为 Samba 用户设定密码时用来存放密码的，详细的 smbpasswd 命令将在 13.2.2 节中讲述。

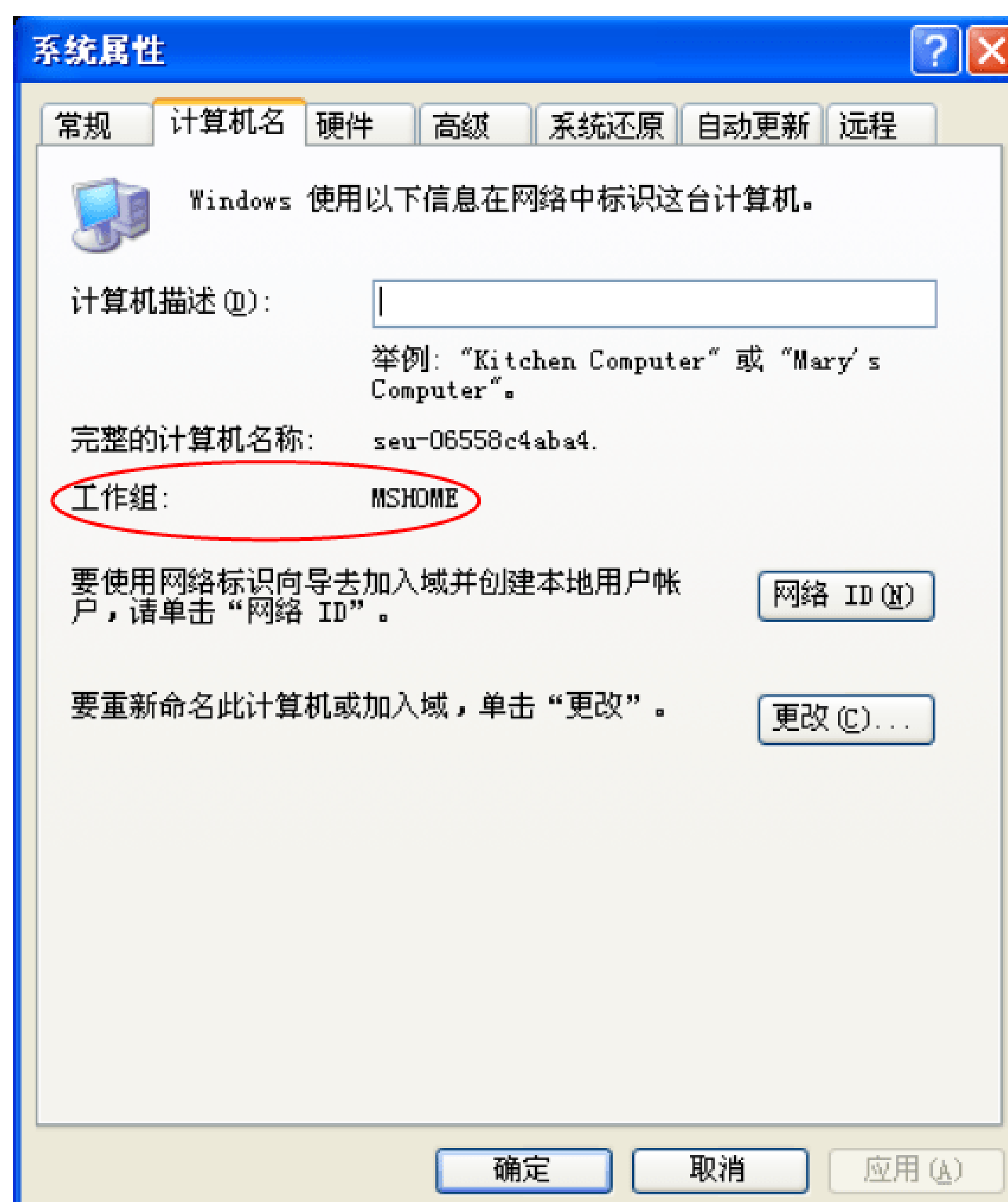


图 13-4　查看 Windows XP 主机的群组名

```
# You may wish to use password encryption. Please read
# ENCRYPTION.txt, Win95.txt and WinNT.txt in the Samba documentation.
# Do not enable this option unless you have read those documents
  encrypt passwords = yes
  smb passwd file = /etc/samba/smbpasswd
```

图 13-5　设定密码文件

❸ 如果需要利用 Samba 服务共享打印机，在全局设定部分找到 printcap name 和 load printers

两行，将注释符去掉，第一行表示加载打印机配置文件的路径，第二行表示 Samba 服务启动时就自动加载所有打印机列表，如图 13-6 所示。

```
# if you want to automatically load your printer list rather
# than setting them up individually then you'll need this
  printcap name = /etc/printcap
  load printers = yes
```

图 13-6　共享打印机设置

❹ 全局设定完毕后，选择[homes]标签，进行 Samba 用户根目录的权限设定，这里有三个设定项，如图 13-7 所示。第一项，"comment"是解释项，没有实际用处。第二项，"browseable=yes 或 no"，表示是否允许用户浏览整个/home 目录，如果"browseable=yes"则 Samba 用户可以浏览所有 Linux 系统用户的根目录，因此安全起见，一般将 browseable 设为 no。第三项，"writable=yes 或 no"，表示是否允许对 Samba 用户根目录进行写操作，场景中要求不允许，因此设为 no。

```
#============================ Share Definitions =================
[homes]
   comment = Home Directories
   browseable = no
   writable = no
```

图 13-7　根目录设置

❺ 最后设定共享目录部分，选择[public]标签，默认是以";"注释，去掉注释符号，共有六行设定项，如图 13-8 所示。第一项，comment 是解释项，没有实际用处。第二项，path 表示所要共享的目录，场景中要求为/home/sharing 目录。第三项，"public=yes"表示开放上面 path 中指定的目录。第四项和第五项相关联，用于设定读写权限，第四项表示是否是只读的，场景中要求/home/sharing 目录为可写，那么设"read only=yes"；第五项表示不是所有用户都有可写权限的情况下有可写权限的用户的列表，由于场景中要求该目录对任何用户都可写，所以我们将该行注释符号去掉。第六项，"writable=yes 或 no"表示是否允许登录用户对该共享目录拥有可写权限，按照场景要求，将 writable 设为 yes。

当然，我们还需要创建/home/sharing 目录，使用下面命令创建，如图 13-9 所示。

```
mkdir /home/sharing
```

```
# A publicly accessible directory, but read only,
# the "staff" group
[public]
   comment = Public Stuff
   path = /home/sharing
   public = yes
   read only = no
;  write list = @staff
   writable = yes
```

图 13-8　开放共享目录

```
[root@sec samba]# mkdir /home/sharing
[root@sec samba]# ll -l /home/sharing
总计 0
```

图 13-9　创建共享目录

❻ 全局设定完毕后，使用 testparm 命令对 smb.conf 进行语法测试，检查是否存在语法错误，输入 testparm 命令，如图 13-10 所示，提示按 Enter 键显示 smb.conf 中所设定的信息。按 Enter 键后，屏幕上显示了全局设定、打印机设定、根目录设定及共享目录设定的信息，如图 13-11 所示。

```
[root@sec samba]# testparm
Load smb config files from /etc/samba/smb.conf
Processing section "[homes]"
Processing section "[printers]"
Processing section "[public]"
Loaded services file OK.
WARNING: lock directory /var/cache/samba should have permissions 0755 for browsing to work
WARNING: passdb expand explicit = yes is deprecated
Server role: ROLE_STANDALONE
Press enter to see a dump of your service definitions
```

图 13-10　测试 smb.conf

```
[global]
        workgroup = MSHOME
        server string = MSHOME Samba Server
        log file = /var/log/samba/%m.log
        max log size = 50
        socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192
        printcap name = /etc/printcap
        dns proxy = No
        hosts allow = 172.18.12.
        cups options = raw

[homes]
        comment = Home Directories
        browseable = No

[printers]
        comment = All Printers
        path = /var/spool/samba
        printable = Yes
        browseable = No

[public]
        comment = Public Stuff
        path = /home/sharing
        read only = No
        guest ok = Yes
[root@sec samba]#
```

图 13-11　smb.conf 设定结果

点评与拓展：smb.conf 除了[homes]、[public]等目录权限设定外，还提供了临时目录、私有目录等更灵活的设定方案，从而使 Samba 服务器可以更细粒度地控制对外共享的目录及每个目录的权限。

13.2.2　Samba 服务器端用户设定

Samba 服务使用 Linux 系统用户登录，但是系统用户登录系统时的密码保存在/etc/passwd 文件中。由于 Samba 服务无法读取/etc/passwd 文件中的密码，所以需要另外建

立一个密码文件保存 Samba 用户的密码。Samba 服务所带的 smbpasswd 命令就是用于建立这样一个密码文件的，对于每一个需要登录 Samba 服务器的系统用户，都要使用 smbpasswd 命令为其建立一个密码文件。

❶ 在 Samba 服务安装之初，列出/etc/samba 目录，有三个文件，当然包括 smb.conf 主设定文件在内，如图 13-12 所示。

```
[root@sec samba]# ls
lmhosts  smb.conf  smbusers
```

图 13-12 /etc/samba 目录内容

❷ 场景中要求 Samba 服务器使用 Linux 系统用户 tian 登录，假设 tian 用户已经存在(如果不存在，读者可使用 adduser 命令添加)，我们使用下面命令为 tian 用户添加 Samba 服务器的密码，如图 13-13 所示。

```
smbpasswd -a tian
```

```
[root@sec samba]# smbpasswd -a tian
New SMB password:
Retype new SMB password:
startsmbfilepwent_internal: file /etc/samba/smbpasswd did not exist. File successfully created.
account_policy_get: tdb_fetch_uint32 failed for field 1 (min password length), returning 0
account_policy_get: tdb_fetch_uint32 failed for field 2 (password history), returning 0
account_policy_get: tdb_fetch_uint32 failed for field 3 (user must logon to change password), retur
account_policy_get: tdb_fetch_uint32 failed for field 4 (maximum password age), returning 0
account_policy_get: tdb_fetch_uint32 failed for field 5 (minimum password age), returning 0
account_policy_get: tdb_fetch_uint32 failed for field 6 (lockout duration), returning 0
account_policy_get: tdb_fetch_uint32 failed for field 7 (reset count minutes), returning 0
account_policy_get: tdb_fetch_uint32 failed for field 8 (bad lockout attempt), returning 0
account_policy_get: tdb_fetch_uint32 failed for field 9 (disconnect time), returning 0
account_policy_get: tdb_fetch_uint32 failed for field 10 (refuse machine password change), returnin
Added user tian.
```

图 13-13 为 tian 用户添加密码

输入命令后，两次提示输入密码，接着，如果是第一次执行 smbpasswd 命令就会自动创建/etc/samba/smbpasswd 文件，并将 tian 用户的 Samba 密码以加密的方式写入所创建的 smbpasswd 文件，最后提示创建 tian 用户成功。

执行完 smbpasswd 命令后，我们再列出/etc/samba 目录中的所有文件时，发现多了密码文件 smbpasswd，如图 13-14 所示。

```
[root@sec samba]# ls
lmhosts  secrets.tdb  smb.conf  smbpasswd  smbusers
[root@sec samba]# 
```

图 13-14 生成/etc/samba/smbpasswd 文件

❸ 场景还要求 Windows XP 上的用户 winsamba 具有访问 Linux 服务器/home/sharing 目录的权限，这就需要将 Windows 上所创建的系统用户与 Linux 的系统用户建立映射关系。打开

/etc/samba/smb.conf 文件，在全局设定部分找到 username map 这一行，默认是注释行，将注释符号去掉，后面设定值不变，如图 13-15 所示。这个设定项表示 Samba 服务器读取/etc/samba/smbusers 这个文件中的内容作为用户名间的映射关系，所以只要通过设定/etc/samba/smbusers 来建立 Windows 用户与 Linux 用户的映射关系即可。

```
# Unix users can map to different SMB User names
   username map = /etc/samba/smbusers
```

图 13-15　设定用户映射文件

打开/etc/samba/smbusers 文件，将需要映射的用户添加到该文件中，这个文件的格式为：“Linux 系统用户=映射到的 Windows 用户”。添加下面语句，如图 13-16 所示。

```
tian = winsamba
```

此语句表示 Windows 系统所创建的 winsamba 用户与 Linux 系统的 tian 用户具有一样的权限，即具有访问 Linux 服务器/home/sharing 目录的权限。

```
[root@sec samba]# vi smbusers
# Unix_name = SMB_name1 SMB_name2 ...
root = administrator admin
nobody = guest pcguest smbguest

tian = winsamba
```

图 13-16　建立 Linux 用户与 Windows 用户的映射关系

❹ 还记得/etc/samba/smb.conf 文件中关于 encrypt passwords 的设定么？将 encrypt passwords 设为 yes 时，就采用了加密方式将用户密码传送到 Samba 端，那么 Windows 客户端如何知道要以加密传输，又如何加密传输呢？这就需要 Samba 专门为 Windows 客户端提供注册表文件来实现。Samba 在/usr/share/doc/samba-3.0.21b/registry 目录下存放了提供给各种不同版本的 Windows 客户端的注册表文件，如图 13-17 所示，其中标注的那个文件是提供给 Windows XP 的，只要将这些注册表文件复制到相应的 Windows 客户端，再运行就可以了。

```
[root@sec ~]# ls /usr/share/doc/samba-3.0.21b/registry
FolderRedir.adm                         Win98_PlainPassword.reg
NT4-Locking.reg                         Win9X-CacheHandling.reg
NT4_PlainPassword.reg                   WindowsTerminalServer.reg
Win2000_PlainPassword.reg               WinME_PlainPassword.reg
Win-2Kx-XPP-DeleteCachedProfiles.reg    Win-NT-DeleteRoamingProfile.reg
Win-2Kx-XPP-ForceLocalProfile.reg       WinXP_PlainPassword.reg
Win95_PlainPassword.reg
[root@sec ~]#
```

图 13-17　提供给 Windows 的注册表文件

📖点评与拓展：Samba 服务的设定是由主设定文件/etc/samba/smb.conf 进行控制的，故以上所有的设定都是围绕主设定文件展开的。

13.2.3 Samba 服务的启动

Samba 服务器端配置完毕后，即可启动 Samba 服务。

❶ 使用下面命令可以启动 Samba 服务，如图 13-18 所示，在 13.1 节 “Samba 服务概述” 中知道 Samba 是由 NetBIOS 和 SMB 两个协议同时支持的，因此 Samba 服务也有两个，即 NMB 和 SMB，从图中看到，两个服务都成功启动。关闭、重启 smb 服务，查看 smb 服务状态等命令格式亦列于下方。

```
/etc/init.d/smb start
/etc/init.d/smb  {restart|reload|condrestart|status}
```

```
[root@sec samba]# /etc/init.d/smb start
启动 SMB 服务：                                                        [确定]
启动 NMB 服务：                                                        [确定]
```

图 13-18 启动 Samba 服务

❷ 每次更改/etc/samba/smb.conf 文件后，都需要重启 Samba 服务使设置生效，使用下面命令重新启动 Samba 服务，如图 13-19 所示。

```
/etc/init.d/smb restart
```

```
[root@sec samba]# /etc/init.d/smb restart
关闭 SMB 服务：                                                        [确定]
关闭 NMB 服务：                                                        [确定]
启动 SMB 服务：                                                        [确定]
启动 NMB 服务：                                                        [确定]
```

图 13-19 重启 Samba 服务

13.3 Samba 客户端的配置

应用实例导航——Samba 客户端的配置和测试

※场景呈现

A 公司对 Samba 服务器端配置完毕后，需要分别对局域网内的 Linux 和 Windows 两种不同的客户端进行配置，使其能够连接 Samba 服务器访问共享目录。

※技术要领

(1) Linux 客户端的配置。
(2) Windows 客户端的配置。

13.3.1　Linux 客户端的设置

在服务器端将 Samba 服务配置完毕后，本节介绍 Samba 客户端如何连接 Samba 服务器访问共享目录，首先讲述 Linux 客户端的设置过程。

❶ 登录 172.18.12.178 的 Linux 主机，首先查看是否安装有 Samba 客户端组件，如图 13-20 所示，输入 rpm 命令后，图 13-20 中标注的那个组件就是 Samba 客户端组件。

```
[root@seugrid2 local]# rpm -qa | grep samba
samba-client-3.0.23c-2
samba-3.0.23c-2
system-config-samba-1.2.35-1.1
samba-common-3.0.23c-2
```

图 13-20　查看 Samba 客户端组件

❷ Samba 客户端组件提供的 smbclient 命令可以列出目标主机的共享资源列表，及 Samba 服务器设置的相关信息，命令格式如下：

```
smbclient -L //主机名或 IP 地址 -U 登录用户名
```

本例中为：

```
smbclient -L //172.18.12.179 -U tian
```

输入 smbclient 命令后，shell 提示输入密码，如图 13-21 所示。正确输入密码后，屏幕列出目标主机的共享资源信息，如图 13-22 所示。其中显示的一个共享目录为 public，即上节在 [public]标签中所设定的/home/sharing 目录；另一个为 tian 目录，即登录用户 tian 的根目录。图 13-22 第二个标注处显示了 Samba 服务器的工作组名，即为 MSHOME。

```
[root@seugrid2 local]# smbclient -L //172.18.12.179 -U tian
Password:
```

图 13-21　smbclient 命令

❸ smbclient 命令也同时提供了登录 Samba 服务器、访问共享目录的功能，命令格式为：

```
smbclient //主机名或 IP 地址/共享目录名 -U 登录用户名
```

本例中为：

```
smbclient -L //sec/tian -U tian
```

正确输入密码后，就可以登录到 tian 用户的根目录/home/tian，如图 13-23 所示。可以像在本地一样输入 shell 命令，图中输入 ls 命令列出了目录内容。

```
[root@seugrid2 local]# smbclient -L //172.18.12.179 -U tian
Password:
Domain=[SEC] OS=[Unix] Server=[Samba 3.0.21b-2]

        Sharename       Type      Comment
        ---------       ----      -------
        ADMIN$          IPC       IPC Service (MSHOME Samba Server)
        IPC$            IPC       IPC Service (MSHOME Samba Server)
        public          Disk      Public Stuff
        tian            Disk      Home Directories
Domain=[SEC] OS=[Unix] Server=[Samba 3.0.21b-2]

        Server               Comment
        ---------            -------

        Workgroup            Master
        ---------            -------
        MSHOME               DFTALENT
[root@seugrid2 local]#
```

图 13-22　显示 Samba 设置结果

```
[root@seugrid2 local]# smbclient //sec/tian -U tian
Password:
Domain=[SEC] OS=[Unix] Server=[Samba 3.0.21b-2]
smb: \> ls
  .                                   D        0  Sat Oct 27 12:55:17 2007
  ..                                  D        0  Sat Nov  3 15:44:55 2007
  .globus                            DH        0  Tue Apr  3 17:05:55 2007
  .bash_profile                       H      191  Tue Apr  3 17:04:17 2007
  .bash_logout                        H       24  Tue Apr  3 17:04:17 2007
  .rnd                                H     1024  Tue Apr  3 17:05:43 2007
  .bashrc                             H      124  Tue Apr  3 17:04:17 2007
  .bash_history                       H       64  Tue Apr  3 17:19:56 2007
  sshrrunon                           A      220  Sat Oct 27 11:37:51 2007
  .gtkrc                              H      120  Tue Apr  3 17:04:17 2007
  sshrrun                             A      112  Sat Oct 27 11:37:51 2007
  hostgen                             A      176  Sat Oct 27 11:37:51 2007
  .zshrc                              H      658  Tue Apr  3 17:04:17 2007
  .kde                               DH        0  Tue Apr  3 17:04:17 2007
  .emacs                              H      515  Tue Apr  3 17:04:17 2007

                65219 blocks of size 1048576. 48065 blocks available
smb: \>
```

图 13-23　访问 Samba 服务器端/home/tian 目录

4. 使用下面命令访问 Samba 服务器的/public 目录测试用户权限，即/home/sharing。

```
smbclient -L //sec/public -U tian
```

由于 public 目录开放了写权限，因此可以新建目录，输入下面命令新建 sambadir 目录:

```
mkdir sambadir
```

如图 13-24 所示，新建 sambadir 成功，说明 public 目录确实对用户开放了可写权限。

5. Samba 服务的日志文件存放在/var/log/samba 目录中，工作组 MSHOME 中的所有主机都建立了日志文件，如图 13-25 所示，其中标注的文件 seugrid2.log 即为上述 172.18.12.178 主机的日志文件。查看 seugrid2.log 后面 40 行可以发现，日志文件记录了我们访问 tian 和 public 目录的历史，如图 13-26 标注部分所示。

```
[root@seugrid2 local]# smbclient //sec/public -U tian
Password:
Domain=[SEC] OS=[Unix] Server=[Samba 3.0.21b-2]
smb: \> ls
  .                                   D        0  Sat Nov  3 19:07:38 2007
  ..                                  D        0  Sat Nov  3 15:44:55 2007

                65219 blocks of size 1048576. 48065 blocks available
smb: \> mkdir sambadir
smb: \> ls
  .                                   D        0  Sat Nov  3 19:07:55 2007
  ..                                  D        0  Sat Nov  3 15:44:55 2007
  sambadir                            D        0  Sat Nov  3 19:07:55 2007

                65219 blocks of size 1048576. 48065 blocks available
smb: \>
```

图 13-24　测试 public 权限

```
[root@sec samba]# ls
172.18.12.155.log  172.18.12.238.log    nmbd.log              seugrid2.log
172.18.12.178.log  ef024a9e0d464b3.log  seu-06558c4aba4.log   smbd.log
[root@sec samba]#
```

图 13-25　所有 Samba 服务的日志文件

```
[root@sec samba]# tail -n 40 seugrid2.log
[2007/11/03 13:41:13, 1] smbd/service.c:make_connection_snum(693)
  seugrid2 (172.18.12.178) connect to service tian initially as user tian (uid=503, gid=503) (pid 26623)
[2007/11/03 15:34:46, 1] smbd/service.c:close_cnum(885)
  seugrid2 (172.18.12.178) closed connection to service tian
[2007/11/03 19:02:25, 1] smbd/service.c:make_connection_snum(693)
  seugrid2 (172.18.12.178) connect to service tian initially as user tian (uid=503, gid=503) (pid 32737)
[2007/11/03 19:03:54, 1] smbd/service.c:close_cnum(885)
  seugrid2 (172.18.12.178) closed connection to service tian
[2007/11/03 19:04:02, 1] smbd/service.c:make_connection_snum(693)
  seugrid2 (172.18.12.178) connect to service public initially as user tian (uid=503, gid=503) (pid 32759
[2007/11/03 19:04:14, 1] smbd/service.c:close_cnum(885)
  seugrid2 (172.18.12.178) closed connection to service public
[2007/11/03 19:04:46, 1] smbd/service.c:make_connection_snum(693)
  seugrid2 (172.18.12.178) connect to service public initially as user tian (uid=503, gid=503) (pid 325)
```

图 13-26　seugrid2.log 日志内容

点评与拓展：使用 smbclient 命令进入 Samba 服务的共享目录后，命令的使用方式与 FTP 客户端相似，下载文件的命令是 get，上传文件的命令是 put，由 help 命令提供帮助信息。

13.3.2　Windows 客户端的设置

介绍了 Linux 客户端如何连接 Samba 服务器之后，接着讲述 Windows 客户端如何连接 Samba 服务器的共享目录。

❶ 在桌面上双击【网上邻居】图标，弹出如图 13-27 所示的【网上邻居】对话框，单击左侧

【网络任务】下拉框中的【查看工作组计算机】命令，如图 13-27 标注部分所示，弹出如图 13-28 所示的窗口。右侧主窗口列出了所有 MSHOME 工作组的计算机，其中有一个名为 MSHOME Samba Server (Sec)，这个主机名正是我们在/etc/samba/smb.conf 文件的全局设定部分 server string 设定项的内容。

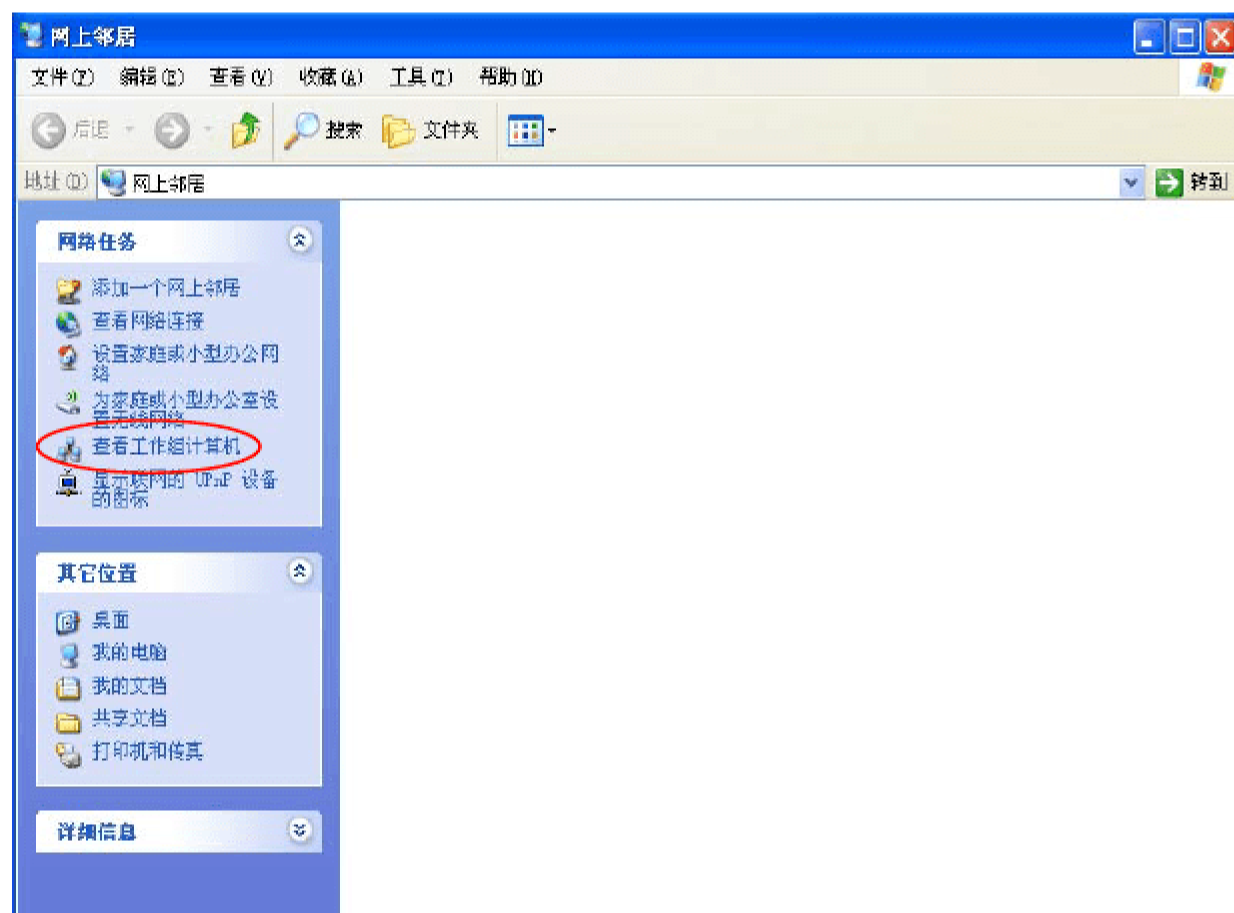


图 13-27 【网上邻居】对话框

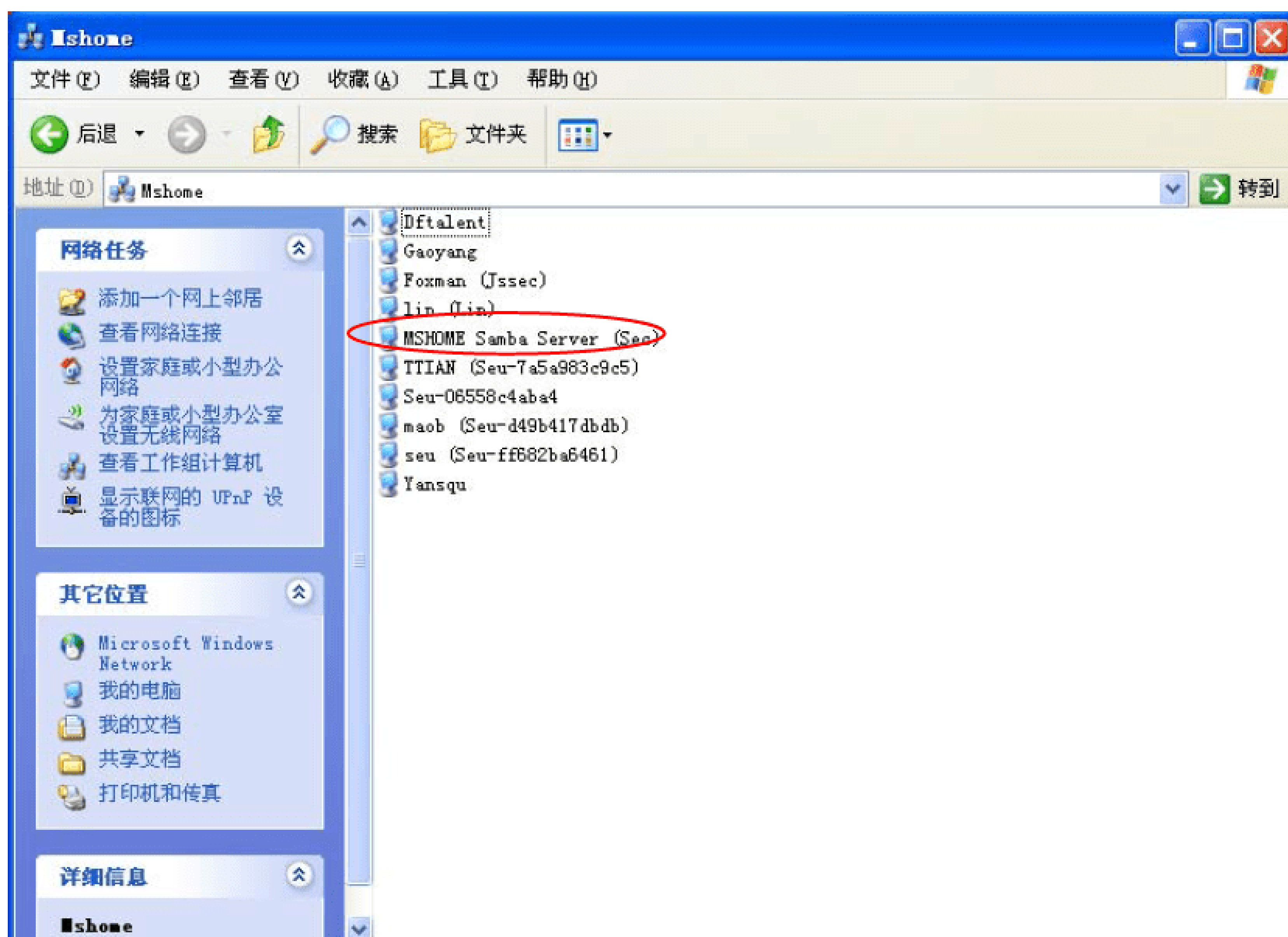


图 13-28 MSHOME 工作组计算机列表

❷ 双击 MSHOME Samba Server (Sec)图标连接该 Samba 服务器，弹出提示输入用户名和密码的对话框，如图 13-29 所示，本例中输入 tian 用户，也可以在 Windows XP 下新建 winsamba

用户进行登录。用户名/密码验证成功后，弹出如图 13-30 所示的窗口，其中列出了我们开放的 public 和 tian 两个共享目录，以及开放的打印机目录，地址栏中显示的是“\\Sec”，这正是 Samba 服务器 172.18.12.179 的主机名。

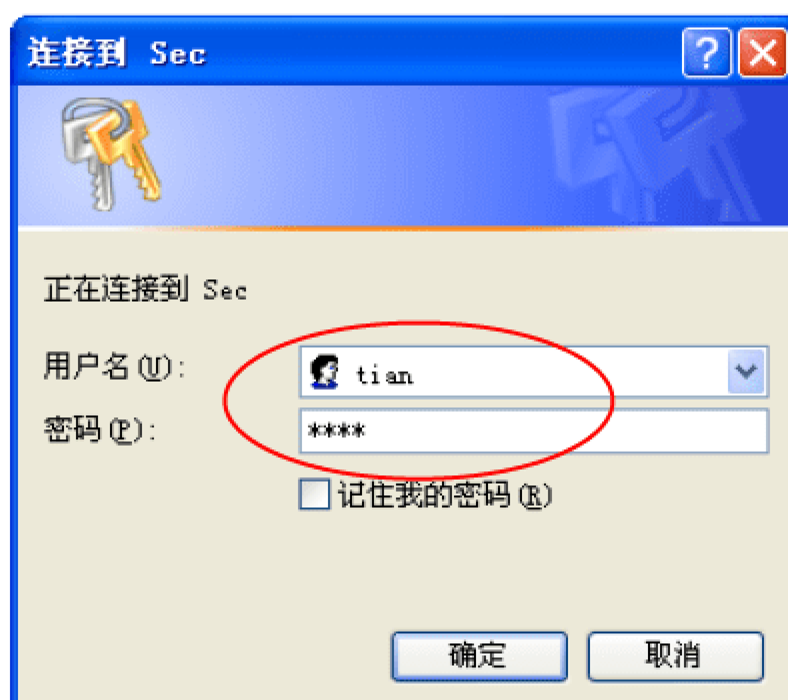


图 13-29　Samba 服务器登录窗口

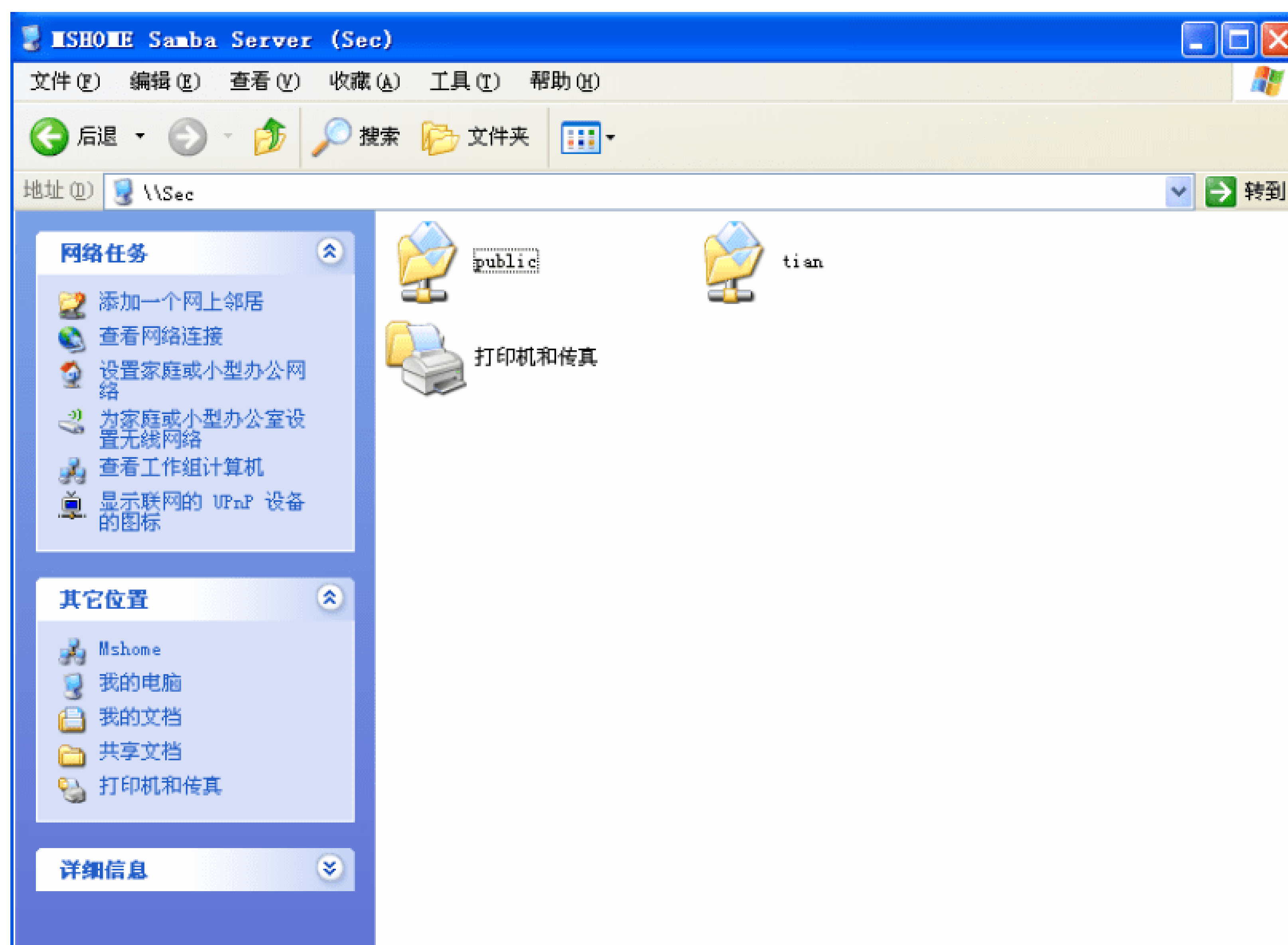


图 13-30　Samba 服务器共享目录列表

❸ 进入 Samba 服务器的 public 目录，可以看到里面已经有了我们在 Linux 客户端远程创建的 sambatest 目录，在 Windows 客户端同样可以在 public 目录下创建目录或文件，我们示例性地新建一文本文件，如图 13-31 所示。

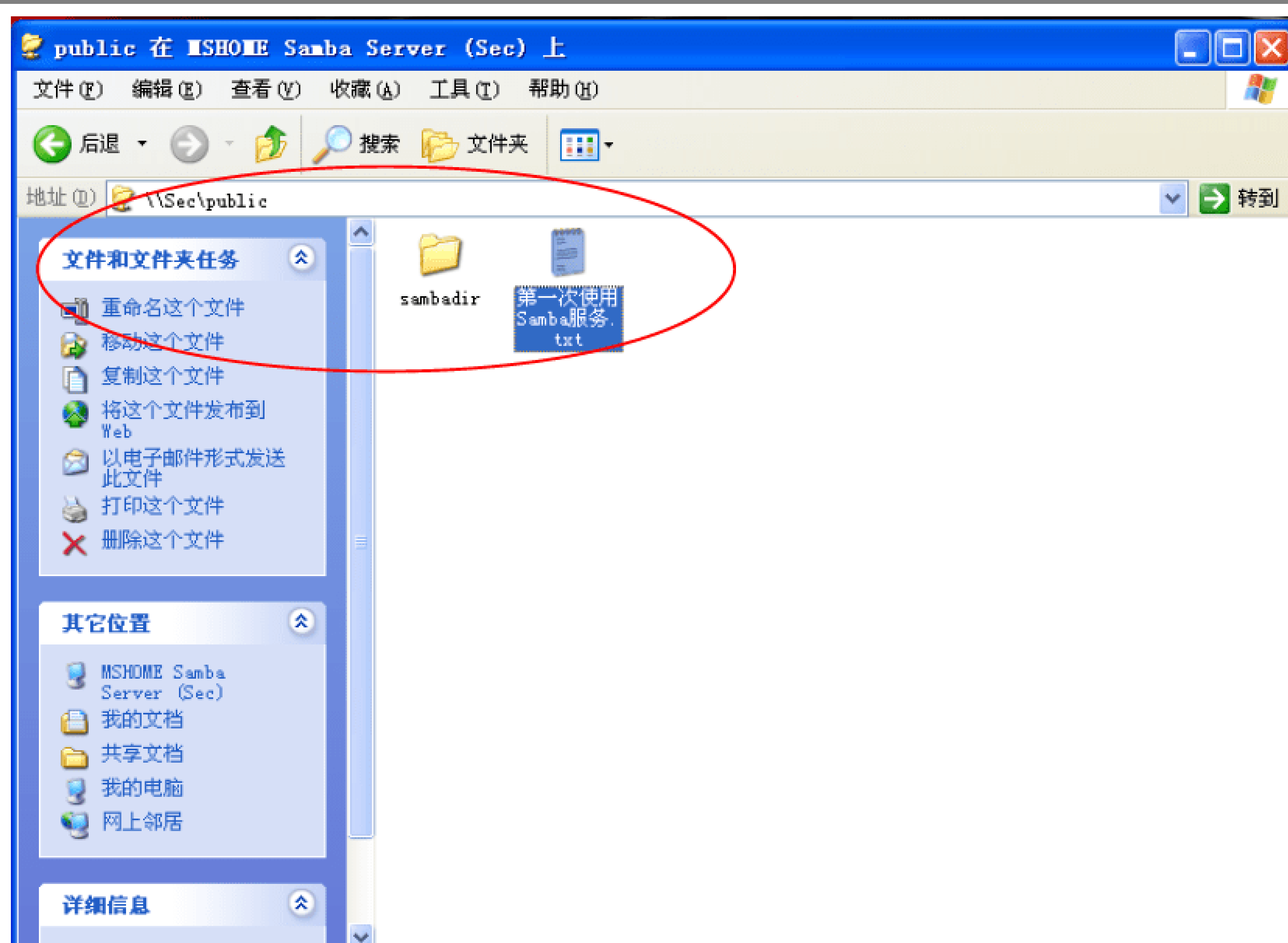


图 13-31　在 public 中新建文件

❹ 从 Windows 系统成功登录后，我们同样可以在 Samba 服务器端的/var/log/samba 目录中查看 Windows 客户端的日志文件，文件名为 seu-06558c4aba4.log 的文件即为上述 Windows 客户端的日志文件，查看其最后 10 行发现它记录了 Windows 客户端的 IP 地址和访问历史，如图 13-32 所示。

```
[root@sec samba]# pwd
/var/log/samba
[root@sec samba]# tail -n 10 seu-06558c4aba4.log
[2007/11/03 19:27:20, 1] smbd/service.c:close_cnum(885)
  seu-06558c4aba4 (172.18.12.140) closed connection to service public
[2007/11/03 19:27:21, 1] smbd/service.c:make_connection_snum(693)
  seu-06558c4aba4 (172.18.12.140) connect to service public initially as user tian (uid=503, gid=50
[2007/11/03 19:27:21, 1] smbd/service.c:close_cnum(885)
  seu-06558c4aba4 (172.18.12.140) closed connection to service public
[2007/11/03 19:27:21, 1] smbd/service.c:make_connection_snum(693)
  seu-06558c4aba4 (172.18.12.140) connect to service public initially as user tian (uid=503, gid=50
[2007/11/03 19:27:21, 1] smbd/service.c:close_cnum(885)
  seu-06558c4aba4 (172.18.12.140) closed connection to service public
[root@sec samba]#
```

图 13-32　查看 Windows 客户端的日志文件

📖**点评与拓展**：在 Windows【开始】菜单的【运行】对话框中输入“\\Samba服务器名”或其 IP 地址同样可以访问 Samba 服务器的共享目录。



13.4　本 章 小 结

Samba 服务对于网络资源共享、多操作系统之间的互联具有重要意义，虽然其服务仅限于在信任的局域网环境内提供打印机共享、文件共享等，但它仍然是配置局域网的实用技术之一。

本章首先介绍了 Samba 服务器的概念及其工作原理，重点介绍了 Samba 服务器端的配置方法及其用户权限管理，最后分别介绍了 Linux 和 Windows 两种客户端如何连接 Samba 服务器访问共享目录。

第 14 章　网络时间服务器的配置与架设

NTP(Network Time Protocol，网络时间协议)，主要用于 Linux 服务器按全球标准时间(UTC)的网络时间校正，以满足各类时间敏感型应用的需求。本章介绍 NTP 服务器的概念及其存在的意义，着重介绍架设一台位于第三层的 NTP 服务器的配置方法，最后分别介绍 Linux 和 Windows 两种客户端如何使用 NTP 服务器进行网络校时。

通过本章的学习，读者应掌握以下内容：

✧　NTP 服务的概念和意义
✧　NTP 服务器端的配置
✧　NTP 客户端的配置

14.1　NTP 服务概述

14.1.1　NTP 服务概述

计算机的系统时间是由石英晶体振荡电路以固定的振荡频率产生的，一般来说，由于晶振不可避免的存在少许误差，导致所产生的系统时间与全球标准时间(UTC)之间存在少许偏差，随着时间的推移，偏差会越来越大。这种越来越大的偏差会给一些时间敏感度较高的网络应用带来不可预知的灾难，如银行交易系统、电子商务系统及网络数据库存储系统等，这些应用对操作的时间都要求比较精确。

NTP 正是为了解决这种问题而产生的，NTP 协议最早由美国 Delaware 大学的 Mills 教授于 1982 年提出，1992 年发布了 NTP v3，这也是当前的主流协议版本，2001 年又发布了最新的 NTP v4 版。

NTP 是使计算机系统与 NTP 服务器或精确时钟源同步化的一种协议，提供了高精度的网络时间校正。精确时钟源是所有 NTP 服务器获取准确时间的源头，它通过原子钟、天文台、卫星或 Internet 等多种方式获得全球标准时间，然后层次式地向下面的 NTP 服务器传播。

NTP 将所有 NTP 服务器按照离精确时钟源的距离，分为不同的层次(stratum)，从 Internet 角度看，我们将 NTP 服务器分为两类：一层(Stratum 1)的 NTP 服务器，它是指有全球标准时间接入的服务器，这些一层服务器是所有 NTP 服务器时间的源头，通常设置了访问权限，一般只提供给二层服务器访问；二层(Stratum 2)的 NTP 服务器，它从一层 NTP 服务器获取全球标准时间，并对外提供开放服务，任何下层的 NTP 服务器可以通过访问二层服务器获

得全球标准时间，并且任何个人服务器也可以自愿加入二层服务器池，为广大用户提供 NTP 服务。官方网站 http://www.pool.ntp.org 提供了二层 NTP 服务器资源的加入和查询世界各地的二层 NTP 服务器地址的功能，第一时间显示了目前活跃的二层 NTP 服务器，比如：目前非洲仅有一台二层 NTP 服务器，中国有四台活跃的 NTP 服务器。三层(Stratum 3)及三层以下的服务器就是用户自行架设的 NTP 服务器，它通过二层服务器获得全球标准时间，向局域网或某个 IP 网段提供标准时间校正服务。层数取决于它从哪层获取全球标准时间，源头为精确时钟源的 NTP 服务器为第一层，以此类推，但是总层数被限制在 15 层以内。

NTP 协议基于 RFC 1305/1119 实现，是能够运用于广域网的大规模同步时间协议，可以获得毫秒级的精度。NTP 协议的精妙之处在于它中间的算法可以从一个或多个时间源，通过时间戳，考虑网络延迟、抖动等服务质量(Quality of Service，QoS)参数的计算得出精确时间。

那么 NTP 客户端又是如何与 NTP 服务器进行时间同步的呢？其工作流程如图 14-1 所示，简述如下：

(1) NTP 客户端向服务器端发送一个请求进行时间同步的 UDP 包，该包记录了它离开客户端时的时间戳。

(2) NTP 服务器接收到该包后，记录包到达服务器的时间戳及包离开客户端的时间戳等数据，然后立即把包返回给客户端。

(3) NTP 客户端接收到服务器的回应后，记录包返回到客户端的时间戳，然后通过 NTP 协议提供的算法计算出客户端与服务器之间的时间偏移量。

(4) NPT 客户端利用时间偏移量调整本地时钟，使其时间与服务器保持同步。

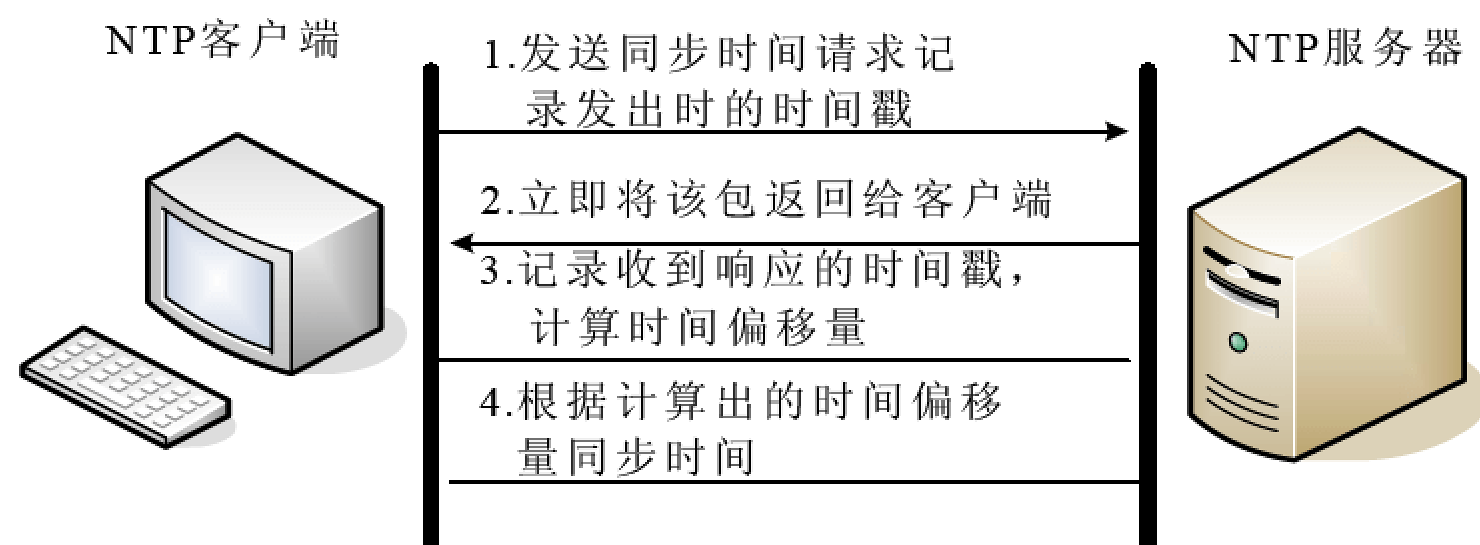


图 14-1　NTP 工作流程

14.1.2　NTP 协议组件简介

Fedora 6.0 已经默认安装了 NTP 协议，主要包含以下两个组件：

✧ ntp

NTP 服务器的主要组件，包括了所有的配置文件和执行文件。

✧ tzdata

NTP 服务器的附属组件，提供了各个时区对应的显示格式，即 Time Zone Data。

NTP 服务器的配置文件主要包含以下几个。

- ✧ /etc/ntp.conf
  NTP 服务守护进程的主配置文件，下节将详细介绍其配置。
- ✧ /usr/share/zoneinfo/
  该目录是 Linux 系统提供的，而不是 NTP 所提供的。这个目录下的文件规定了世界各主要时区的时间设定。
- ✧ /etc/sysconfig/clock
  该文件是 Linux 的主要时区设定文件，每次开机后系统自动读取该文件来设定自己系统所预设要显示的时区，具体时区文件就是到上面的/usr/share/zoneinfo/目录中相应地读取。
- ✧ /etc/localtime
  该文件记录了本地时间，规定了系统的默认时区，通常是复制/usr/share/zoneinfo/相应的时区文件生成。
- ✧ /bin/date
  Linux 系统输入和修改日期与时间的命令。
- ✧ /sbin/hwclock
  写入或读取 BIOS 时间的命令，必须是 root 权限方能执行该命令。由于 Linux 系统的 BIOS 时间与系统时间是分开的，所以使用 date 命令调整了时间之后，还需要使用 hwclock 才能将修改过的时间写入 BIOS 中。
- ✧ /usr/sbin/ntpd
  NTP 服务守护进程的入口程序。
- ✧ /usr/sbin/ntpdate
  客户端进行时间同步的命令。
- ✧ /usr/sbin/ntptrace
  用于追踪某台时间服务器的时间对应关系的命令，可以测试 NTP 服务器。

14.2 NTP 服务器端的配置

应用实例导航——架设 NTP 服务器

※场景呈现

为某内部局域网架设 NTP 服务器，所架设的 NTP 服务器为第三层 NTP 服务器，对外公开访问，从第二层服务器获得全球标准时间。

※技术要领

(1) NTP 服务器端的配置。
(2) NTP 服务器端的启动和测试。

本节介绍 NTP 服务器端的配置方法，重点介绍 NTP 服务主配置文件/etc/ntp.conf 的意义和设定方法；然后介绍如何启动 NTP 服务器，以及如何测试 NTP 服务器。

14.2.1　NTP 服务器端设定

NTP 服务的主配置文件/etc/ntp.conf 设置了该 NTP 服务器的外部时钟源的地址以及其他主机访问的权限。下面介绍/etc/ntp.conf 的意义和设定方法。

❶ 首先，输入下面命令确认 ntp 组件是否安装，如图 14-2 所示，目前 Linux 系统的 ntp 版本一般为 4.2.*。

```
rpm -qa | grep ntp
```

```
[root@sec phpMyAdmin]# rpm -qa | grep ntp
chkfontpath-1.10.1-1
ntp-4.2.0.a.20050816-11
```

图 14-2　检查 NTP 是否安装

❷ NTP 服务主配置文件/etc/ntp.conf 已经有了一些默认的设定，而且以“#”号开头的那些语句行对大部分的默认设定语句都进行了注释，因此，我们只需根据需要去掉注释或添加几行即可。打开/etc/ntp.conf 文件，如图 14-3 所示。

```
restrict default nomodify notrap noquery
```

此行说明不允许其他计算机修改或查询配置在本机上的 NTP 服务，其中 default 参数表示所有 IP 地址。

```
[root@sec etc]# vi ntp.conf
# Permit time synchronization with our time source, but do not
# permit the source to query or modify the service on this system.

restrict default nomodify notrap noquery

# Permit all access over the loopback interface.  This could
# be tightened as well, but to do so would effect some of
# the administrative functions.
restrict 127.0.0.1


# -- CLIENT NETWORK -------
# Permit systems on this network to synchronize with this
# time service.  Do not permit those systems to modify the
# configuration of this service.  Also, do not use those
# systems as peers for synchronization.
# restrict 192.168.1.0 mask 255.255.255.0 nomodify notrap


# --- OUR TIMESERVERS -----
server 0.asia.pool.ntp.org
server 1.asia.pool.ntp.org
server 2.asia.pool.ntp.org
```

图 14-3　/etc/ntp.conf 设定

```
restrict 127.0.0.1
```

此行说明开放本机内部环路接口，用于在本地对 NTP 服务器进行监控和测试。

如果要对访问该 NTP 服务器的客户端进行限制，也是使用 restrict 语句。假如所架设的 NTP 服务器只对内部局域网开放，局域网 IP 范围为 192.168.10.0，那么添加下面语句，如图 14-4 所示：

```
restrict 192.168.10.0 mask 255.255.255.0 nomodify notrap
```

```
# -- CLIENT NETWORK -------
# Permit systems on this network to synchronize with this
# time service.  Do not permit those systems to modify the
# configuration of this service.  Also, do not use those
# systems as peers for synchronization.
 restrict 192.168.10.0 mask 255.255.255.0 nomodify notrap
```

图 14-4　限制客户端 IP 范围

由于本例中要求该 NTP 服务器对外面的所有主机都开放，因此，这段也采用默认设置即可。

由注释行“# --- OUR TIMESERVERS -----”开始的段落，设定了上层的 NTP 服务器，由于我们所架设的 NTP 服务器处于第三层，因此这里应该添加第二层的 NTP 服务器。可以通过查询http://www.pool.ntp.org网站，发现中国目前活跃的第二层 NTP 服务器地址有 server 0.asia.pool.ntp.org 等四个，如图 14-5 标注部分所示，因此，将这几个活跃第二层 NTP 服务器的地址写入/etc/ntp.conf 之中，如图 14-3 的标注部分所示。

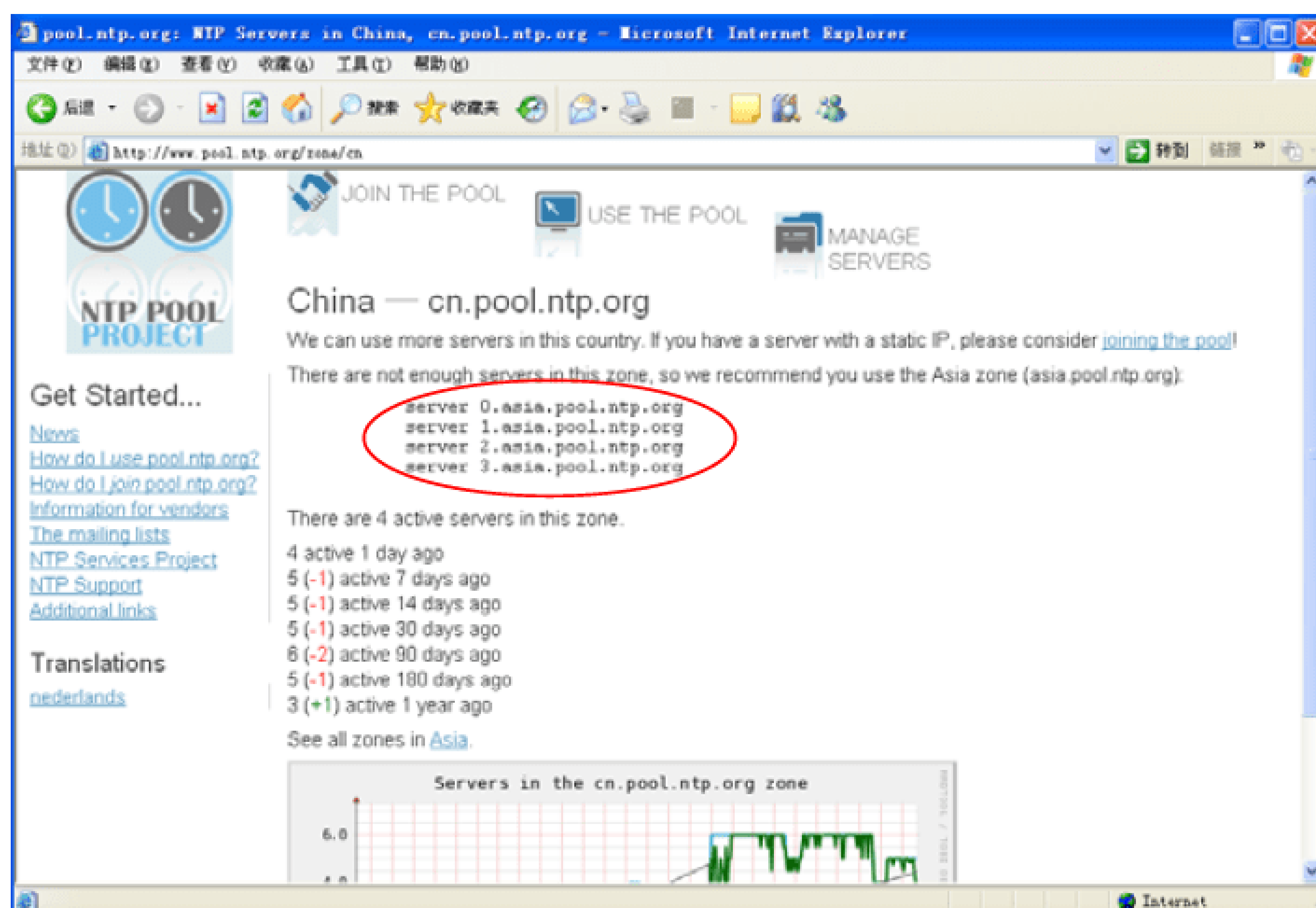


图 14-5　查看中国的 NTP 服务器

❸ /etc/ntp.conf 文件的其他部分采用默认设置即可，下面介绍其中的两个配置项，分别为本地时钟源设定和自然漂移的设定，如图 14-6 所示。

```
server  127.127.1.0     # local clock
fudge   127.127.1.0 stratum 10

#
# Drift file.  Put this in a directory which the daemon can write to.
# No symbolic links allowed, either, since the daemon updates the file
# by creating a temporary in the same directory and then rename()'ing
# it to the file.
#
driftfile /var/lib/ntp/drift
broadcastdelay  0.008
```

图 14-6　本地时钟源和自然漂移设定

```
server  127.127.1.0     # local clock
fudge   127.127.1.0 stratum 10
```

fudge 语句将本地时钟源设置为第 10 层，优先级很低，这样的设定使该服务器一旦能连接外部 NTP 服务器就可以通过外部时钟源进行时钟同步，在确实没有外部时钟源的情况下才使用本地时钟源进行时钟同步。

```
driftfile /var/lib/ntp/drift
```

NTP 守护进程每次与第二层 NTP 服务器连接时，都会自动更新该文件，用于补偿系统时钟的自然漂移，从而使得系统时钟即使在切断与外部时钟源连接的情况下，仍然能够在一段时间内保持相当的精确度。

❹ 最后设定/etc/sysconfig/ntpd 文件，添加 SYNC_HWCLOCK=yes 一行语句，如图 14-7 所示，这样每当该 NTP 服务器通过外部时钟源校正时间后，会自动更新 BIOS 时间，使得系统时间和 BIOS 时间同步。

```
[root@sec etc]# vi /etc/sysconfig/ntpd
# Drop root to id 'ntp:ntp' by default.
OPTIONS="-u ntp:ntp -p /var/run/ntpd.pid"
SYNC_HWCLOCK=yes
~
```

图 14-7　/etc/sysconfig/ntpd 设定

点评与拓展：　NTP 服务器相当于时间代理服务器，通过连接第二层 NTP 服务器获得全球标准时间，再以一定的权限开放给下面的主机，因此重要的设定包括第二层 NTP 服务器的地址和它的开放权限。

14.2.2　NTP 服务的启动和测试

NTP 服务配置完毕后，就可以启动 NTP 服务。

❶ ntpd 服务仍然是在/etc/init.d 目录下，使用下面命令启动，如图 14-8 所示。

```
/etc/init.d/ntpd start
```

关闭、重启 ntpd 服务，查看 ntpd 服务状态等命令格式亦列于下方。

```
/etc/init.d/ntpd  {restart|reload|condrestart|status}
```

然后通过观察网络端口，检查 ntpd 进程是否在某个端口上进行监控，输入下面命令:

```
netstat -tlunp | grep ntp
```

结果如图 14-8 所示，ntpd 服务在本地环路网卡及 eth0 网卡上的 123 号端口进行 UDP 包的监听。

```
[root@sec etc]# /etc/init.d/ntpd start
启动 ntpd:                                              [确定]
[root@sec etc]# netstat -tlunp | grep ntp
udp        0      0 172.18.12.179:123           0.0.0.0:*                               25659/ntpd
udp        0      0 127.0.0.1:123               0.0.0.0:*                               25659/ntpd
udp        0      0 0.0.0.0:123                 0.0.0.0:*                               25659/ntpd
udp        0      0 :::123                      :::*                                    25659/ntpd
```

图 14-8　启动 ntpd 服务

其他重启、关闭 ntpd 的命令类似于其他 Linux 系统服务，图 14-9 演示了重启 ntpd 服务的结果。

```
[root@sec ~]# /etc/init.d/ntpd restart
关闭 ntpd:                                              [确定]
启动 ntpd:                                              [确定]
```

图 14-9　重启 ntpd 服务

❷ ntpd 启动成功后，它就开始尝试与上层的 NTP 服务器进行连接。那么如何确认所架设的 NTP 服务器与其上层的 NTP 服务器连接成功了呢？可以使用 ntpstat 命令来查看，图 14-10 所示的信息表示与上层的 NTP 服务器尚未连接成功。等过了一段时间之后，可以再次输入 ntpstat 命令查看，如图 14-11 所示，发现连接成功，提示为在第三层连接成功，这说明我们所架设的第三层 NTP 服务器已经成功地与第二层的 NTP 服务器连接上，并能进行时间同步。

```
[root@sec etc]# ntpstat
unsynchronised
  time server re-starting
   polling server every 64 s
```

图 14-10　连接第二层 NTP 服务器不成功

```
[root@sec ~]# ntpstat
synchronised to NTP server (220.130.158.71) at stratum 3
   time correct to within 495 ms
   polling server every 64 s
```

图 14-11　连接第二层 NTP 服务器成功

❸ 还可以使用下面的命令查看本机 NTP 服务的同步状态，如图 14-12 所示。

```
ntpq -p
```

该命令列出了可用来校时的上层 NTP 服务器列表。

```
[root@sec etc]# ntpq -p
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
 bbs.med.cgu.edu .INIT.          16 u    - 1024    0    0.000    0.000 4000.00
 starlite.ispwor .INIT.          16 u    - 1024    0    0.000    0.000 4000.00
 163.25.109.18   .INIT.          16 u    - 1024    0    0.000    0.000 4000.00
*LOCAL(0)        LOCAL(0)        10 l    5   64  377    0.000    0.000   0.004
[root@sec etc]#
```

图 14-12　检查 NTP 服务的同步状态

4 如果该 NTP 服务器启动了防火墙，还要注意一下安全设定，否则经常会因为防火墙的设置问题导致无法连接到上层的 NTP 服务器。假设我们需要对内部子网 192.168.10.0 开放，那么就需要添加下面的规则，如图 14-13 所示。

```
iptables -A INPUT -p UDP -i eth1 -s 192.168.10.0/24 --dport 123 -j ACCEPT
```

该命令对内部子网开启端口号为 123 的 UDP 端口，命令参数的含义在此不再详述，读者可以参考第 3 章“Linux 防火墙与 NAT 服务”。

```
[root@sec ~]# iptables -A INPUT -p UDP -i eth1 -s 192.168.10.0/24 --dport 123 -j ACCEPT
[root@sec ~]#
```

图 14-13　NTP 服务器的安全设定

点评与拓展：仍然需要强调的是，启动 ntpd 服务时，shell 提示成功，未必就说明 ntpd 服务正常运行。应该查看其侦听进程，并且测试 ntpd 服务是否与上层的 NTP 服务器成功连接。

14.3　NTP 客户端的配置

应用实例导航——NTP 客户端配置和测试

※场景呈现

为某内部局域网架设 NTP 服务器完毕后，需要分别对局域网内的 Linux 和 Windows 两种不同的客户端进行配置，使其能够成功地通过 NTP 服务器同步更新时间。

※技术要领

(1) Linux 客户端的配置。
(2) Windows 客户端的配置。

14.3.1 Linux 客户端的配置

在服务器端将 NTP 服务配置完毕后，本节介绍 NTP 客户端如何使用 NTP 服务器校正时间，首先讲述 Linux 客户端的设置过程。

❶ Linux 客户端使用 ntpdate 命令进行时间同步，如图 14-14 所示。ntpdate 命令格式如下:

```
ntpdate [-nv] [NTP IP地址/主机名]
```

本例使用 IP 地址确认 NTP 服务器，因此命令为:

```
ntpdate 172.18.12.179
```

通过 ntpdate 校正的是系统时间，我们再将其写入 BIOS，使得系统时间与 BIOS 时间同步，命令如下:

```
hwclock -w
```

顺便介绍一下 date 命令，如果该命令不带任何参数就是读取当前的系统时间，如果后面带如下格式的参数，就是更新当前的系统时间。

```
date MMDDhhmmYYYY
```

其中，MM: 月份；DD: 日期；hh: 小时；mm: 分钟；YYYY: 西元年。

```
[root@seugrid2 ~]# ntpdate 172.18.12.179
23 Oct 20:02:30 ntpdate[23146]: adjust time server 172.18.12.179 offset 0.000416 sec
[root@seugrid2 ~]# date
Tue Oct 23 20:02:32 CST 2007
[root@seugrid2 ~]# hwclock -w
[root@seugrid2 ~]# hwclock -r
Tue 23 Oct 2007 08:02:45 PM CST  -0.117867 seconds
[root@seugrid2 ~]#
```

图 14-14 Linux 客户端进行网络校时

❷ 为了使 Linux 客户端每天都能保持与 NTP 服务器的同步，可以定时执行时间同步命令，只要将其写入 crontab，如图 14-15 标注所示。

```
30 6 * * * root /usr/sbin/ntpdate 172.18.12.179 && /sbin/hwclock -w
```

该语句表明每天的 6: 30 执行 root /usr/sbin/ntpdate 172.18.12.179 和 /sbin/hwclock –w 两条命令进行时间同步。

由于 crontab 在第 16 章 “Linux 服务器的性能监控” 中还会用到，其本身也是 Linux 系统所提供的非常实用的功能，因此，我们详细讲述一下 crontab 相关的知识和命令。crontab 命令的功能是在一定的时间间隔调度一些命令的执行。在/etc 目录下有一个 crontab 文件，这里存放有系统运行的一些调度程序。每个用户可以建立自己的调度 crontab，下面解释几条

crontab 的常用命令:

```
crontab -u user -e 或 vi /etc/crontab
```

```
[root@seugrid2 ~]# vi /etc/crontab
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
HOME=/

# run-parts

30 6 * * * root /usr/sbin/ntpdate 172.18.12.179 && /sbin/hwclock -w
```

图 14-15　每天进行时间同步

用于编辑用户 user 的 cron 作业，用户通过编辑文件来增加或修改任何作业请求。

```
crontab -u user -r
```

用于删除 user 用户的所有的 cron 作业。

```
crontab -u user -l 或 cat /etc/crontab
```

用于显示 user 用户的显示计划。

每个 Linux 系统用户都可以安排自己的计划作业，/etc/cron.allow 表示哪些用户能使用 crontab 命令，如果它是一个空文件表明没有一个用户能安排作业。如果这个文件不存在，而有另外一个文件/etc/cron.deny,则只有不包括在这个文件中的用户才可以使用 crontab 命令。如果它是一个空文件表明任何用户都可安排作业。两个文件同时存在时 cron.allow 优先，如果都不存在，只有超级用户可以安排作业。Linux 系统用户的作业与它们预定的时间储存在文件/usr/spool/cron/crontabs/username 中。username 是用户名，在相应的文件中存放着该用户所要运行的命令。命令执行的结果，无论是标准输出还是错误输出，都将以邮件形式发给用户。

从图 14-15 可以看出 crontab 中的语句格式为:

```
* * * * * 用户名 可执行命令
```

前面的五个“*”分别表示分钟、小时、日期、月份、星期，取值范围分别为 0～59、0～23、1～31、1～12、0～6”，比如:

```
* * * * * //代表每分钟
1 * * * * //代表每小时第 1 分钟
02 12 * * * //代表每天 12 点第 2 分钟(每天 12：02)
0-59/2 * * * * //代表每 2 分钟执行一次任务
```

14.3.2　Windows 客户端的配置

下面讲述 Windows 客户端如何连接 NTP 服务器进行时间同步。

在桌面上选择【我的电脑】→【控制面板】→【日期与时间】命令，弹出如图 14-16

所示的对话框，切换到【Internet 时间】选项卡，在【服务器】文本框中填入刚才所设定的 NTP 服务器地址，本例中为 172.18.12.179，然后单击【立即更新】按钮，提示信息表明同步时间成功。

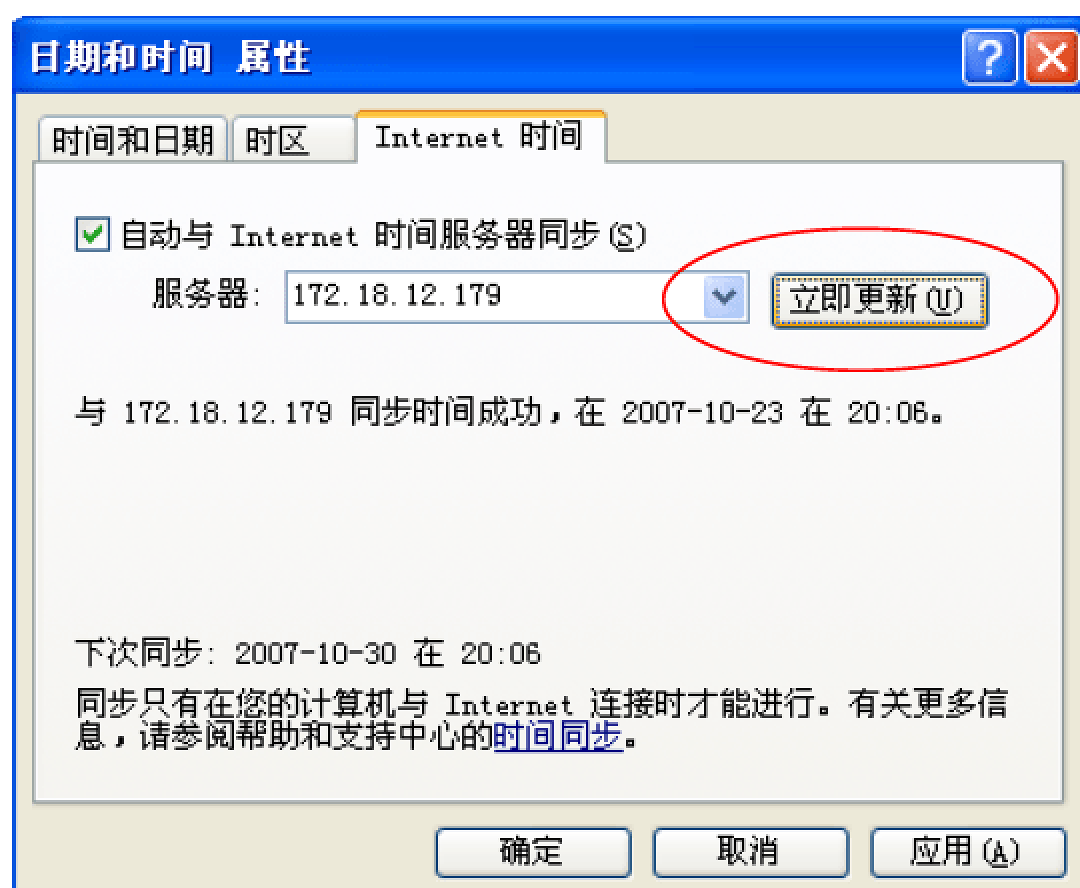


图 14-16　Windows 客户端进行网络校时

点评与拓展：Linux 系统客户端网络校时需要附加设定 BIOS 的时间，在 Windows 中则不需要。

14.4　本 章 小 结

网络时间服务在当今网络中有着非常重要的意义和非常广泛的应用，是所有支持时间敏感型应用所必须架设的服务器。但是就配置和架设 NTP 服务器本身来说并不难，读者可以轻松地理解本章的内容。

本章首先介绍了 NTP 服务器的概念及其存在的意义，重点介绍架设一台位于第三层的 NTP 服务器的配置方法，最后分别介绍 Linux 和 Windows 两种客户端如何使用 NTP 服务器进行网络校时。

第 15 章　使用 Webmin 图形化配置 Linux 服务器

配置 Linux 服务器时往往需要涉及到各种 shell 命令和配置文件，使得初学者头痛不已，如果有一个类似于 Windows 图形窗口的 Linux 图形化管理工具，对于初学者来说无疑是很便利的。Webmin 就是这样一种功能强大、使用方便的 Linux 图形化管理工具。在 Webmin 下，无需使用 shell 命令和手工修改复杂的配置文件，就可以非常方便地对 Linux 服务器进行管理和配置。本章主要介绍怎样利用 Webmin 工具进行 Linux 下各种服务的配置。

通过本章的学习，读者应掌握以下内容：

- ✧ Webmin 的安装和配置
- ✧ 使用 Webmin 配置 DHCP 服务
- ✧ 使用 Webmin 配置 Samba 服务
- ✧ 使用 Webmin 配置 DNS 服务
- ✧ 使用 Webmin 配置 Web 服务
- ✧ 使用 Webmin 配置 NFS 服务
- ✧ 使用 Webmin 配置 SSH 服务
- ✧ 使用 Webmin 配置防火墙服务
- ✧ 使用 Webmin 配置 MySQL 服务
- ✧ 使用 Webmin 管理系统软件

15.1　Webmin 简介

Webmin 是目前功能最强大的基于 Web 的集系统管理和网络管理于一身的图形化 Linux 和 UNIX 的管理工具，网络管理员通过浏览器访问 Webmin 友好的用户窗口，从而可以轻松地管理本地或远程服务器。目前 Webmin 支持绝大多数的 Linux 和 UNIX 系统，这些系统除了各种版本的 Linux 以外还包括 AIX、HPUX、Solaris、UNIXware、Irix 和 FreeBSD 等。

相对于其他 GUI 管理工具而言，Webmin 具有如下显著优点：

- ✧ Web 管理方式使得 Webmin 同时具有本地和远程管理的能力。
- ✧ 插件式结构使得 Webmin 具有很强的扩展性和伸缩性。
- ✧ 访问控制和 SSL 支持为远程管理提供了足够的安全性。
- ✧ 国际化支持，提供多国语言版本，包括简体中文。

Webmin 可以让用户在远程使用支持 HTTPS (SSL 上的 HTTP)协议的 Web 浏览器通过 Web 窗口管理主机，在保证安全性的前提下提供了简单深入的远程管理。这对于系统管理

员来说非常理想，因为所有主流平台都有满足甚至超出上述需求的 Web 浏览器。而且，Webmin 有其自己的“Web 服务器”，不需要运行第三方软件(比如 Web 服务器)。Webmin 的模块化架构允许用户在需要时编写自己的配置模块，本章仅仅介绍前面章节所介绍的重要网络服务器的配置模块，其实 Webmin 还包括许多模块，几乎系统的每一部分都能够通过 Webmin 来配置和管理，更丰富的功能需要读者在实践中挖掘。

15.2 Webmin 的安装和配置

应用实例导航——Webmin 的安装和配置

※场景呈现

A 公司为了方便各类 Linux 服务器的管理，需要利用 Webmin 软件以图形化的方式对各种服务器进行配置、启动和管理。要求为 A 公司安装 Webmin 软件，端口设为 10000 号，登录账户为 root 用户，并且需要在服务器启动时就自动启动 Webmin。

※技术要领

(1) Webmin 的安装。
(2) Webmin 端口和账户的配置。
(3) Webmin 的启动和登录。

Webmin 是 Linux 系统外的第三方软件，因此本节首先介绍 Webmin 的安装，在安装过程中介绍 Webmin 的账户和端口配置；最后介绍如何启动 Webmin，以及如何登录 Webmin 对各类服务器进行管理。

15.2.1 Webmin 的安装

本节介绍 Webmin 的下载和安装过程，具体步骤如下。

1. Webmin 安装包可以到网站 http://prdown.source.net/webadmin 下载，或者从本书的附带光盘中获得。这里演示压缩包的安装方法。本书使用的 Webmin 1.340 版本是目前比较新的版本，提供的功能比较完善，下载窗口如图 15-1 所示。

2. Webmin 安装包名字为 webmin-1.340.tar.gz。在/usr/local 目录下使用下面命令解压安装包，得到/usr/local/webmin-1.340 目录，进入该目录，如图 15-2 所示。

```
tar -zxvf  webmin-1.340.tar.gz
```

安装目录 webmin-1.340 中包含 setup.sh 这个可执行文件，使用下面命令执行之，就开始安装 Webmin，如图 15-3 所示。

```
./setup.sh
```

安装程序首先提示确定 Webmin 的安装路径，默认为/etc/webmin。如果采用默认值，按 Enter 键继续；如果需要改变 Webmin 安装路径，则输入指定的安装路径。接着提示确定 Webmin 的日志目录路径，默认为/var/webmin，一般采用默认值，按 Enter 键继续，如图 15-3 标注部分所示。

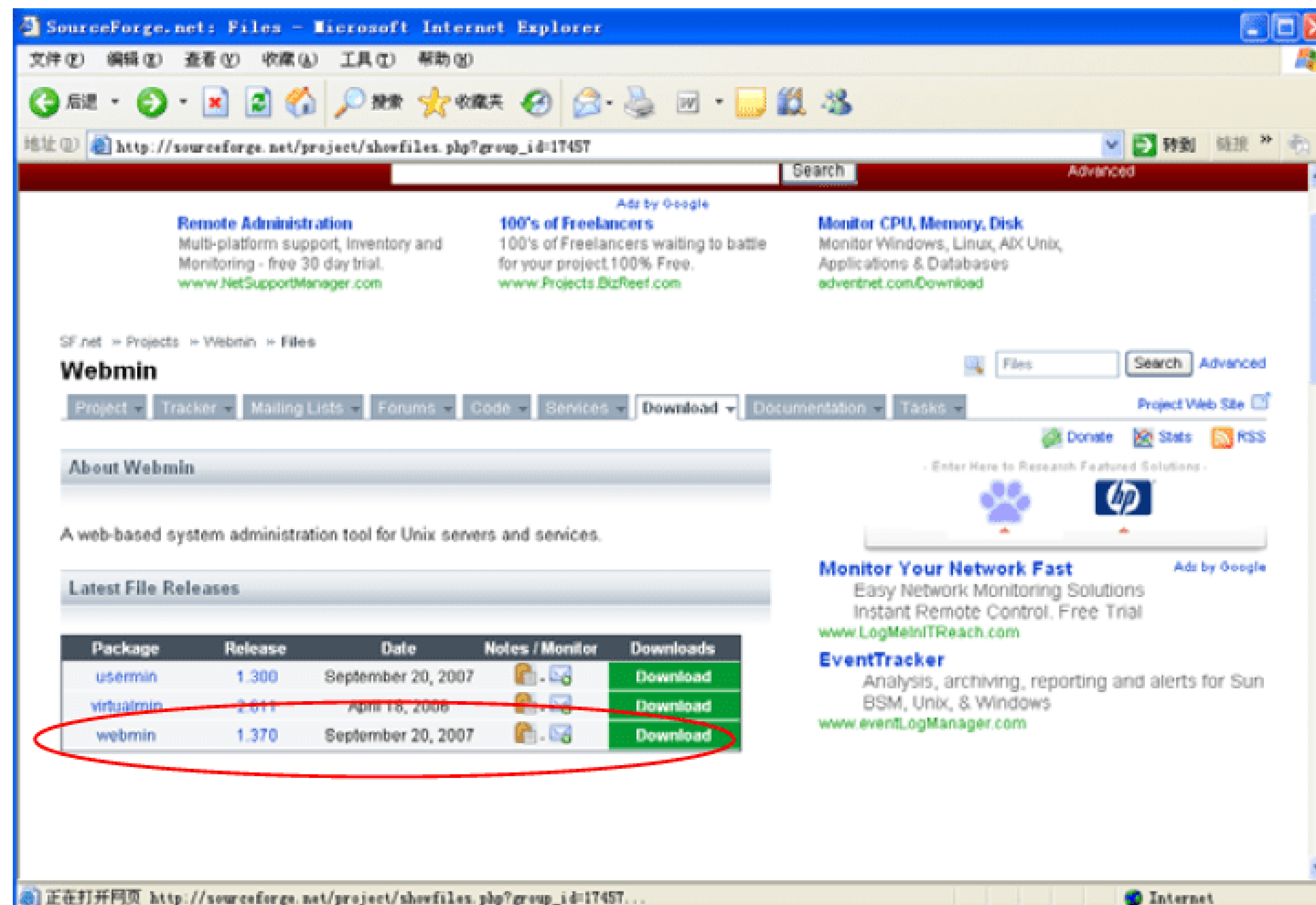


图 15-1　Webmin 下载窗口

```
[root@sec ~]# cd /usr/local/webmin-1.340
[root@sec webmin-1.340]#
```

图 15-2　进入 Webmin 安装目录

```
[root@sec webmin-1.340]# ./setup.sh
***********************************************************************
*            Welcome to the Webmin setup script, version 1.340        *
***********************************************************************
Webmin is a web-based interface that allows Unix-like operating
systems and common Unix services to be easily administered.

Installing Webmin in /usr/local/webmin-1.340 ...

***********************************************************************
Webmin uses separate directories for configuration files and log files.
Unless you want to run multiple versions of Webmin at the same time
you can just accept the defaults.

Config file directory [/etc/webmin]:
Log file directory [/var/webmin]:
```

图 15-3　开始 Webmin 安装

❸ 由于 Webmin 是使用标准的 perl 语言实现的，因此安装程序会提示 perl 语言解释器的路径，默认为/usr/bin/perl。此时务必保证 perl 语言解释器已经安装，且 perl 入口程序路径正确，否则系统将无法安装 Webmin。一般采用默认路径即可，仍然按 Enter 键继续，如图 15-4 标注部分所示。

```
***********************************************************************
Webmin is written entirely in Perl. Please enter the full path to the
Perl 5 interpreter on your system.

Full path to perl (default /usr/bin/perl):

Testing Perl ...
Perl seems to be installed ok

***********************************************************************
```

图 15-4　确认 perl 路径

❹ 接着安装程序提示确认 Webmin 端口，默认的 TCP 端口为 10000 号，如果 10000 端口被其他进程所占用，安装停止，提示信息如图 15-5 所示，这种情况下或者将占用 10000 端口的进程关掉，或者将 Webmin 换成其他空闲端口。端口验证通过后，安装程序提示输入 Webmin 的登录账号和密码，默认为 admin，我们一般使用 root 账户登录，然后重复两次确认密码，如图 15-6 标注部分所示。最后，安装程序询问是否在服务器启动时就启动 Webmin 程序，一般选是。

```
Webmin uses its own password protected web server to provide access
to the administration programs. The setup script needs to know :
 - What port to run the web server on. There must not be another
   web server already using this port.
 - The login name required to access the web server.
 - The password required to access the web server.
 - If the webserver should use SSL (if your system supports it).
 - Whether to start webmin at boot time.

Web server port (default 10000):
ERROR: TCP port 10000 is already in use by another program
```

图 15-5　10000 端口被占用

```
***********************************************************************
Webmin uses its own password protected web server to provide access
to the administration programs. The setup script needs to know :
 - What port to run the web server on. There must not be another
   web server already using this port.
 - The login name required to access the web server.
 - The password required to access the web server.
 - If the webserver should use SSL (if your system supports it).
 - Whether to start webmin at boot time.

Web server port (default 10000):
Login name (default admin): root
Login password:
Password again:
The Perl SSLeay library is not installed. SSL not available.
Start Webmin at boot time (y/n): y
***********************************************************************
```

图 15-6　确认端口和登录账户

所有的安装提示都通过验证后，Webmin 安装成功并成功启动，提示信息如图 15-7 所示。

```
*************************************************************************
Webmin has been installed and started successfully. Use your web
browser to go to

  http://sec:10000/

and login with the name and password you entered previously.
```

图 15-7　安装 Webmin 成功

5. 最后附带介绍卸载 Webmin 的方法，通常不建议手动删除 Webmin 的安装目录，这样常常造成卸载得不干净。我们利用 Webmin 提供的卸载程序卸载，进入 Webmin 安装后的目录，本例为/etc/webmin，其中有一个可执行文件为 uninstall.sh，输入下面命令卸载，如图 15-8 所示。

```
./uninstall.sh
```

```
[root@sec webmin]# pwd
/etc/webmin
[root@sec webmin]# ./uninstall.sh
```

图 15-8　卸载 Webmin

15.2.2　Webmin 的启动

由于上面安装 Webmin 时设置了服务器启动时自动启动 Webmin 程序，因此一般不需手工启动 Webmin，Webmin 启动命令也因此常被忽略。但是，有时在 Webmin 程序被非正常关闭后，却需要 Webmin 的启动命令，因此在此对 Webmin 的启动方法作简单介绍。

1. Webmin 的启动命令仍然在安装目录下，即/usr/local/webmin-1.340 目录，其中的 webmin-init 可执行文件即为 Webmin 的启动命令，通过如下形式调用该命令，如图 15-9 所示。

```
./webmin-init start
```

关闭、重启 Webmin 命令格式如下:

```
./webmin-init {stop|restart}
```

```
[root@sec webmin-1.340]# ./webmin-init start
[root@sec webmin-1.340]# ./webmin-init
Usage: ./webmin-init { start | stop | restart }
[root@sec webmin-1.340]#
```

图 15-9　Webmin 启动命令

2. 如果在安装过程中没有选择在服务器启动时自动启动 Webmin，可以通过如下办法补救：将

Webmin 启动命令写入/etc/rc.local 文件，如图 15-10 标注部分所示，即在服务器启动时就立即执行./webmin-init start 这条命令，这同样可以实现服务器启动时自动启动 Webmin。不过需要注意的是，rc.local 中需要写入 webmin-init 的绝对路径。

```
[root@sec webmin-1.340]# vi /etc/rc.local
#!/bin/sh
#
# This script will be executed *after* all the other init scripts.
# You can put your own initialization stuff in here if you don't
# want to do the full Sys V style init stuff.

touch /var/lock/subsys/local

./usr/local/webmin-1.340/webmin-init start
```

图 15-10　服务器启动立即启动 Webmin

15.2.3　Webmin 的登录

Webmin 正常启动后，就可以通过 Web 浏览器登录 Webmin。

❶ 在 IE 浏览器地址栏输入 http://IP:10000，本例是输入 http://172.18.12.178:10000，如果在本机登录 Webmin 只需输入http://localhost:10000即可。可以看到如图 15-11 所示的登录窗口。

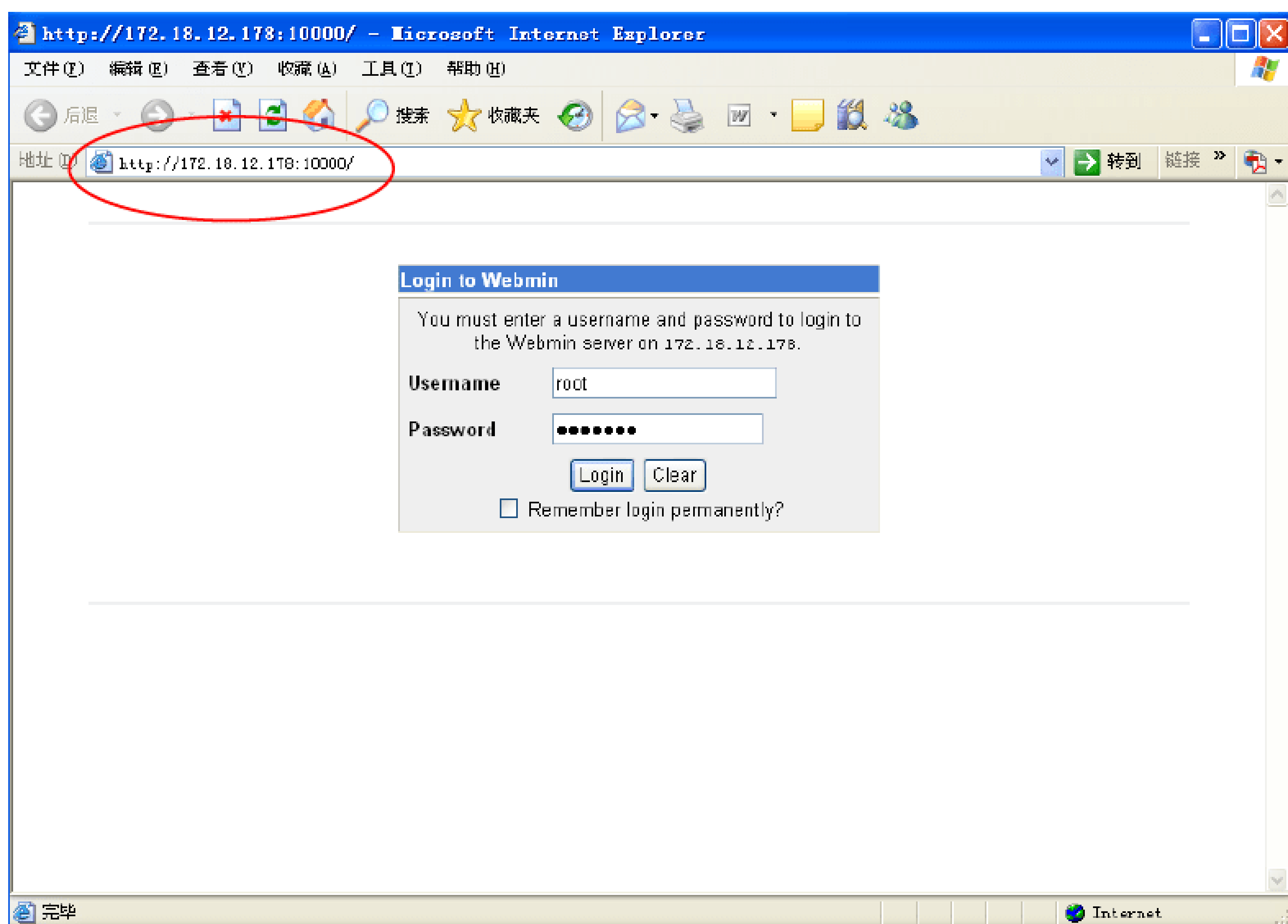


图 15-11　Webmin 登录窗口

❷ 正确输入安装时所设定的用户账户和密码后，即可登录 Webmin，出现如图 15-12 所示的窗口。窗口左侧的菜单即为 Webmin 的管理服务器的初始菜单，如图 15-12 标注部分所示。窗口中部显示了该服务器的性能参数，包括主机名、操作系统版本、Webmin 版本、系统时间、

CPU 平均负载、物理内存、虚拟内存以及磁盘空间等。

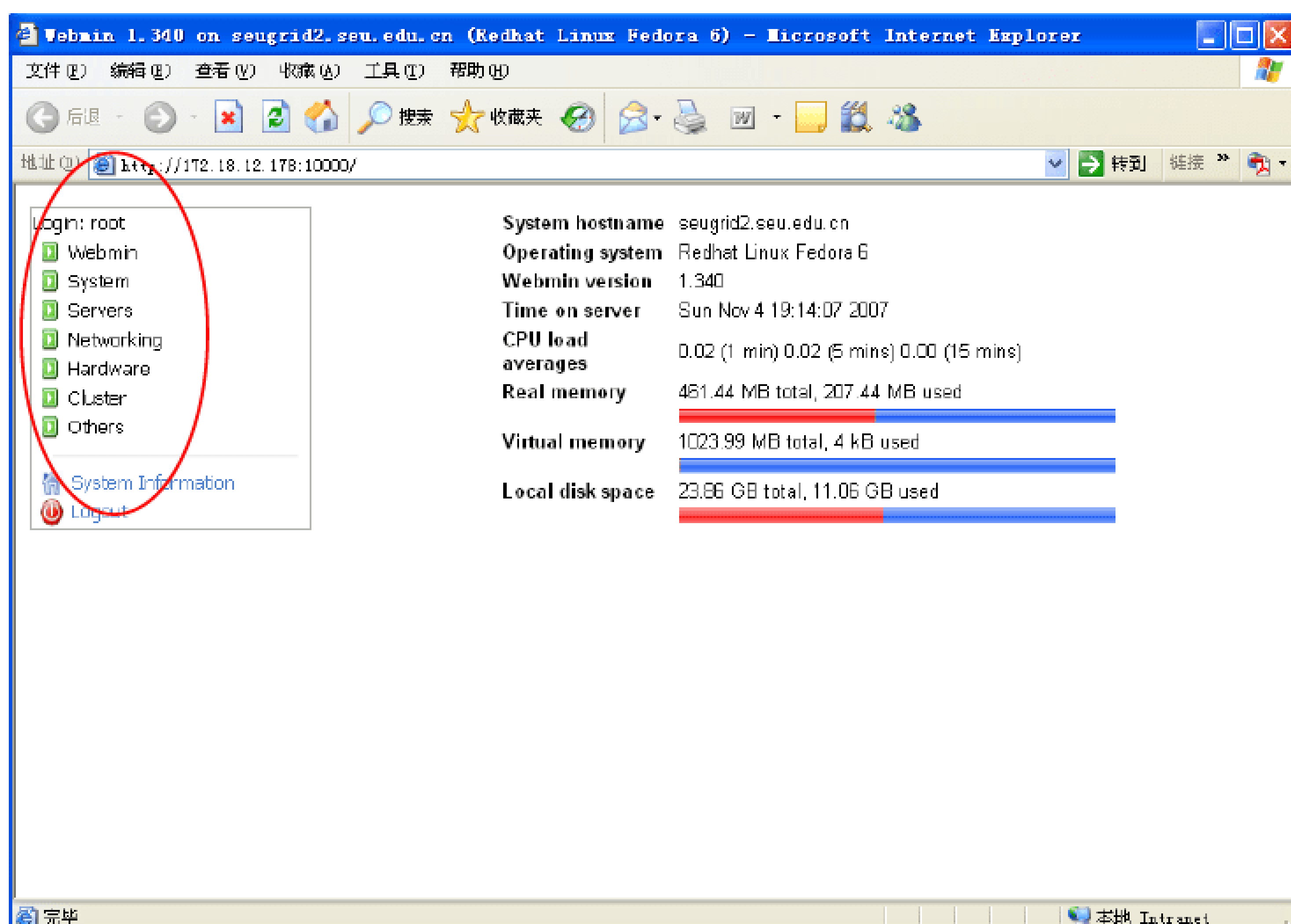


图 15-12　Webmin 初始管理窗口

❸ Webmin 提供了多种语言支持，包括简体中文。在 Webmin 左侧菜单区域选择 Webmin→Webmin Configuration 命令，出现如图 15-13 所示的初始管理窗口。在右侧主窗口的图标区域选择 Language 就进入如图 15-14 所示的设置语言窗口，在 Display in language 下拉列表框中选择语言种类，这里选择 Simplified Chinese (ZH_CN)，即简体中文。然后单击 Change Language 按钮改变窗口语言，就可以出现如图 15-15 所示的中文窗口，但是左侧菜单区域仍然是英文的。

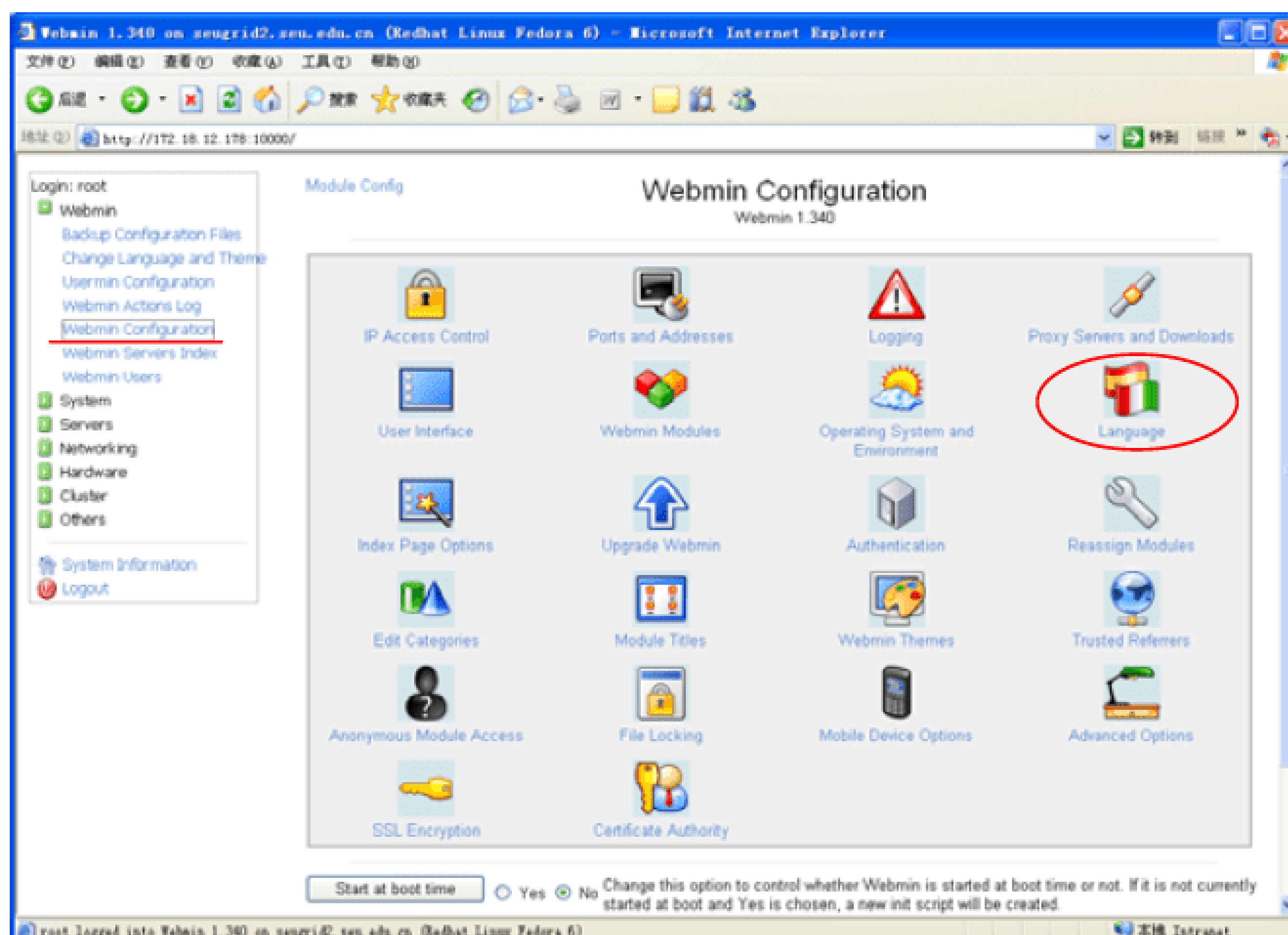


图 15-13　Webmin 初始管理窗口

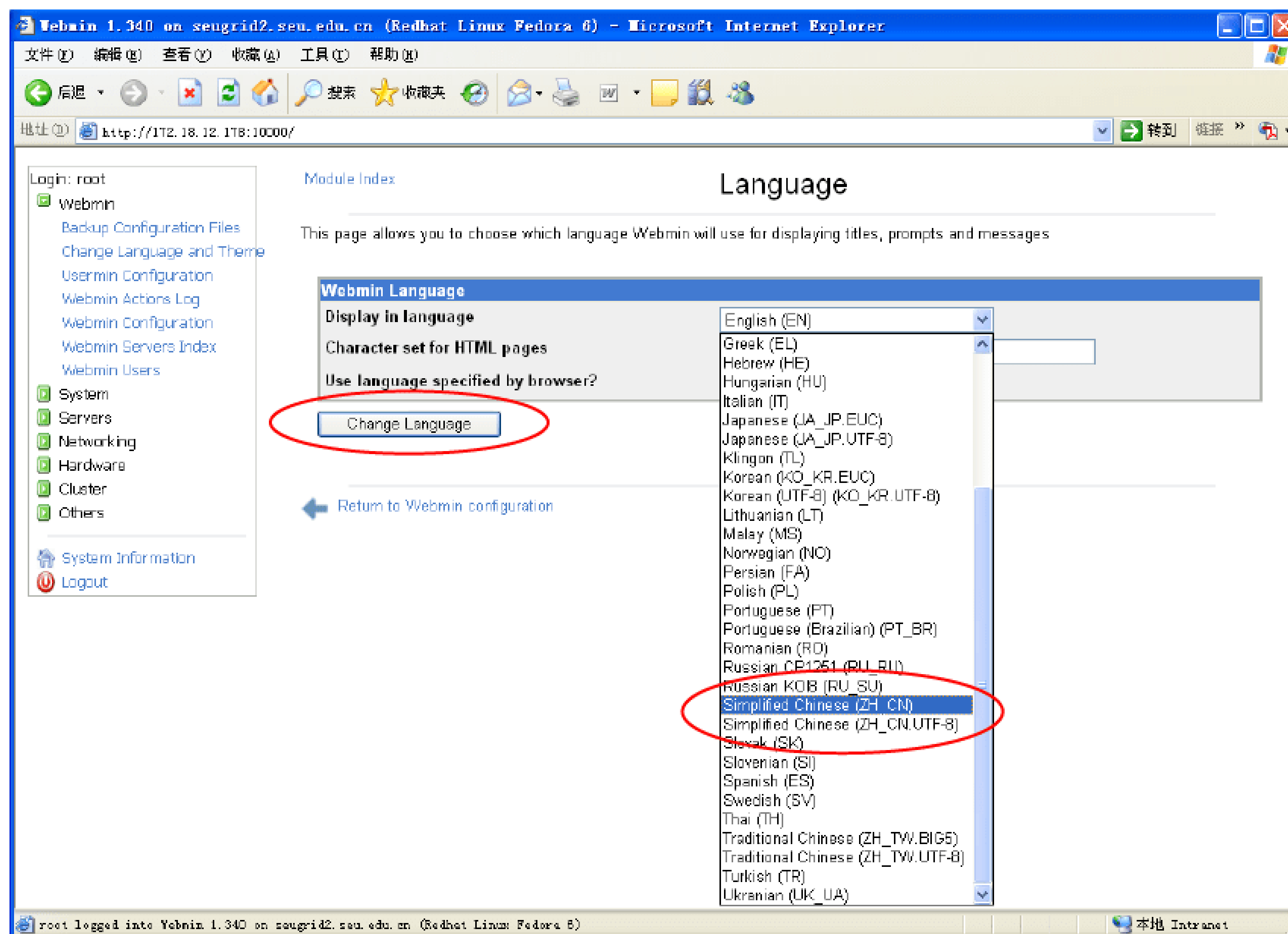


图 15-14　Webmin 设置语言窗口

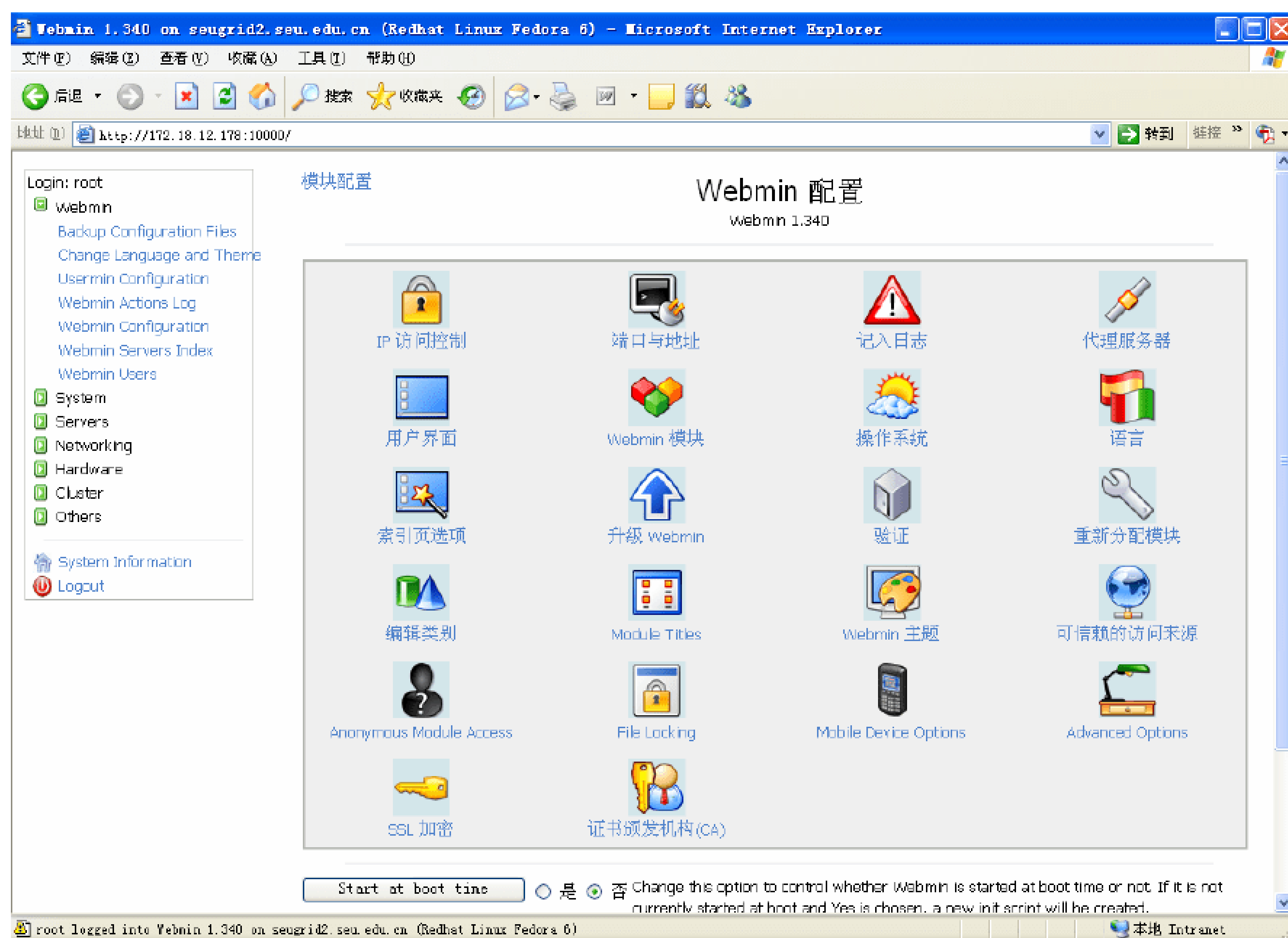


图 15-15　Webmin 中文管理窗口

📖**点评与拓展：**如果 Linux 服务器启动了防火墙，请务必保证 TCP 10000 号端口的畅通，否则将导致 IE 无法远程访问。

15.3　使用 Webmin 配置 DHCP 服务

应用实例导航——利用 Webmin 架设 DHCP 服务器

※场景呈现

A 公司利用 Webmin 配置 DHCP 服务器，要求将 DHCP 服务器动态分配的 IP 地址范围限制在 172.18.12.100~172.18.12.240 之内，其他 IP 地址保留下来；将 seu-06558c4aba4 这台主机静态地与 IP 172.18.12.140 相绑定。另外，DHCP 分配的 IP 的预设有效期是 1 天，最长为 3 天。

假设 A 公司的另一个局域网段为 192.168.10.*，要求为这两个局域子网配置 DHCP 共享网络，同时为这两个网段的主机动态分配 IP 地址。192.168.10.*子网的动态 IP 范围限制为 192.168.10.1～192.168.10.33，IP 的预设有效期同样是 1 天，最长为 3 天。DCHP 共享网络的拓扑示意图如图 15-16 所示。

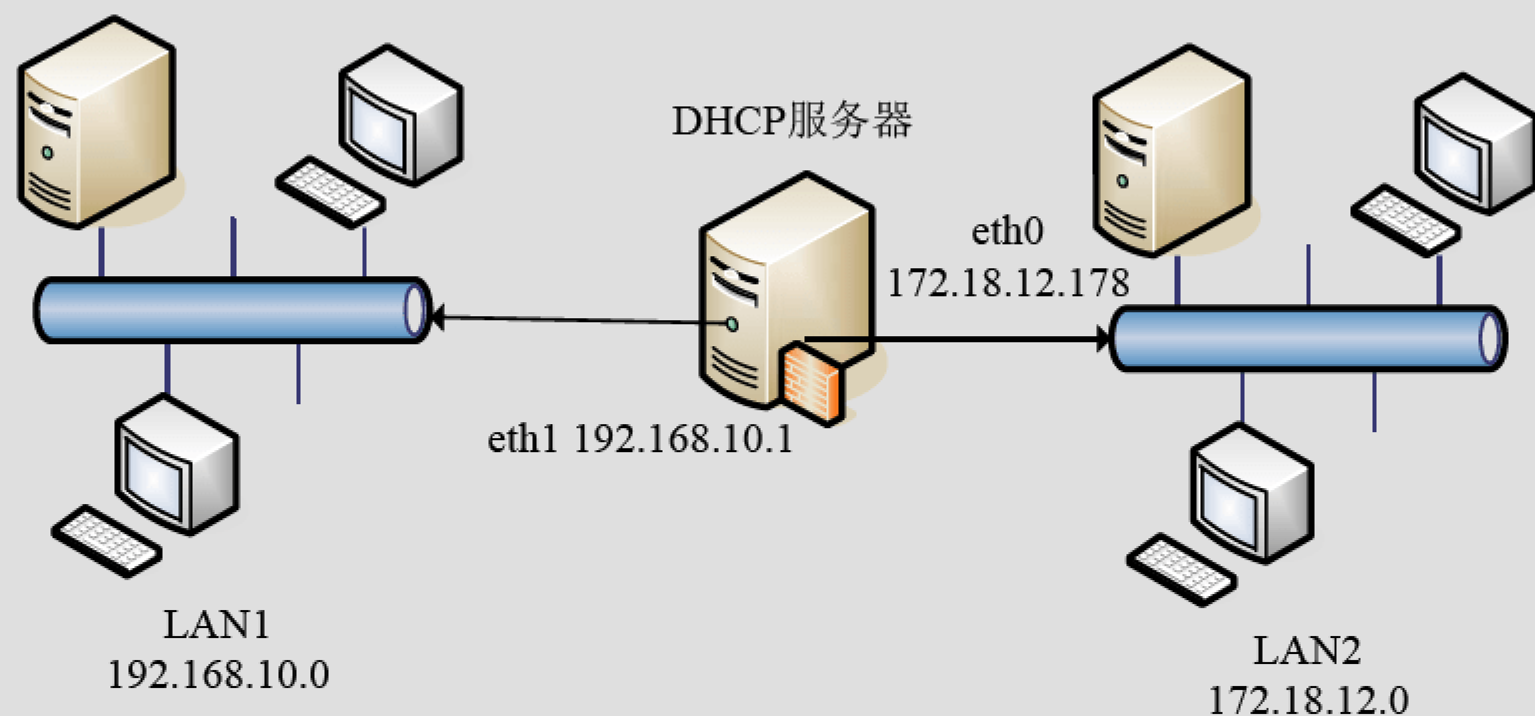


图 15-16　DHCP 共享网络拓扑图

※技术要领

(1) Webmin 配置 DHCP 服务的子网。
(2) Webmin 配置 DHCP 服务的主机。
(3) Webmin 配置 DHCP 共享网络。

从第 6 章“DHCP 服务器的配置与架设”的介绍知道，DHCP 服务用于自动将网络参数

(如 IP 地址、子网掩码、默认网关、DNS 地址等)正确地分配给网络中的每台计算机，使客户端计算机在开机时就自动设定好相关网络参数值。本节讲述如何使用 Webmin 图形化配置 DHCP 服务。Webmin 是在 DHCP 服务安装好的基础上提供图形化的配置和启动功能，DHCP 服务的安装可参考第 6 章的论述，本节从 DHCP 服务安装成功的基础上展开论述。

❶ 在 Webmin 左侧菜单区域选择 Servers→DHCP Server 命令，弹出如图 15-17 所示的 DHCP 服务器配置窗口，其中包括“子网和共享网络”和“主机和主机组”两类配置。



图 15-17　DHCP 服务器配置窗口

❷ 在 DHCP 服务器配置窗口中选择【增加一个新的子网】命令，如图 15-17 标注部分所示，弹出如图 15-18 所示的【创建子网】窗口。场景呈现中要求的局域网段是 172.18.12.*，因此【网络地址】文本框设为 172.18.12.0，【网络掩码】文本框设为 255.255.255.0，【地址范围】文本框设为 172.18.12.100-172.18.12.240，【默认租赁时间】文本框按照要求设为 1 天，即 86400 秒，【最大租赁时间】文本框设为 3 天，即 259200 秒。其他都采用默认设置。单击【新建】按钮创建子网，成功后在 DCHP 服务器配置窗口出现子网图标，如图 15-19 标注部分所示，名字为 172.18.12.0，单击可以查看其配置参数，并可以进入编辑模式修改其配置参数。单击 Delete-Selected 按钮可以删除选中的子网。



图 15-18　【创建子网】窗口

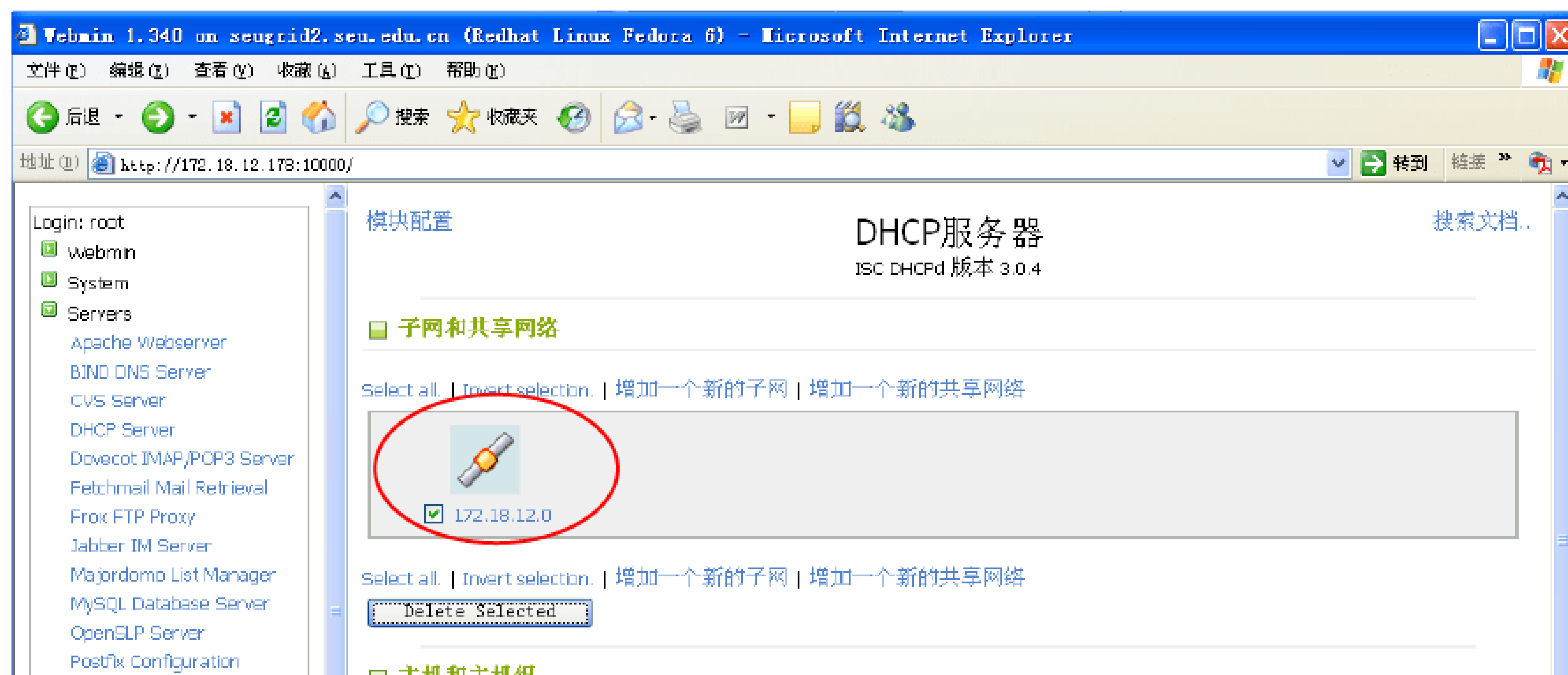


图 15-19　创建子网成功

❸ 在 DHCP 配置窗口下方选择【增加一个新的主机】命令，弹出如图 15-20 所示的创建主机窗口。场景呈现中要求主机名为 seu-06558c4aba4，将其输入【主机名】文本框中；【硬件地址】文本框中输入这台主机的 MAC 地址，这个参数不能输错，因为 DHCP 正是将 IP 地址与 MAC 地址相绑定的。【被指定的主机】文本框中输入所创建的 172.18.12.0 子网，表示将此主机加入到这个子网中；【固定的 IP 地址】文本框按照场景呈现设为 172.18.12.140，表示该主机总是获得该 IP 地址。单击【保存】按钮，创建该主机，成功后在 DCHP 服务器配置窗口下方出现主机图标，如图 15-21 标注部分所示，名字为 seu-06558c4aba4。

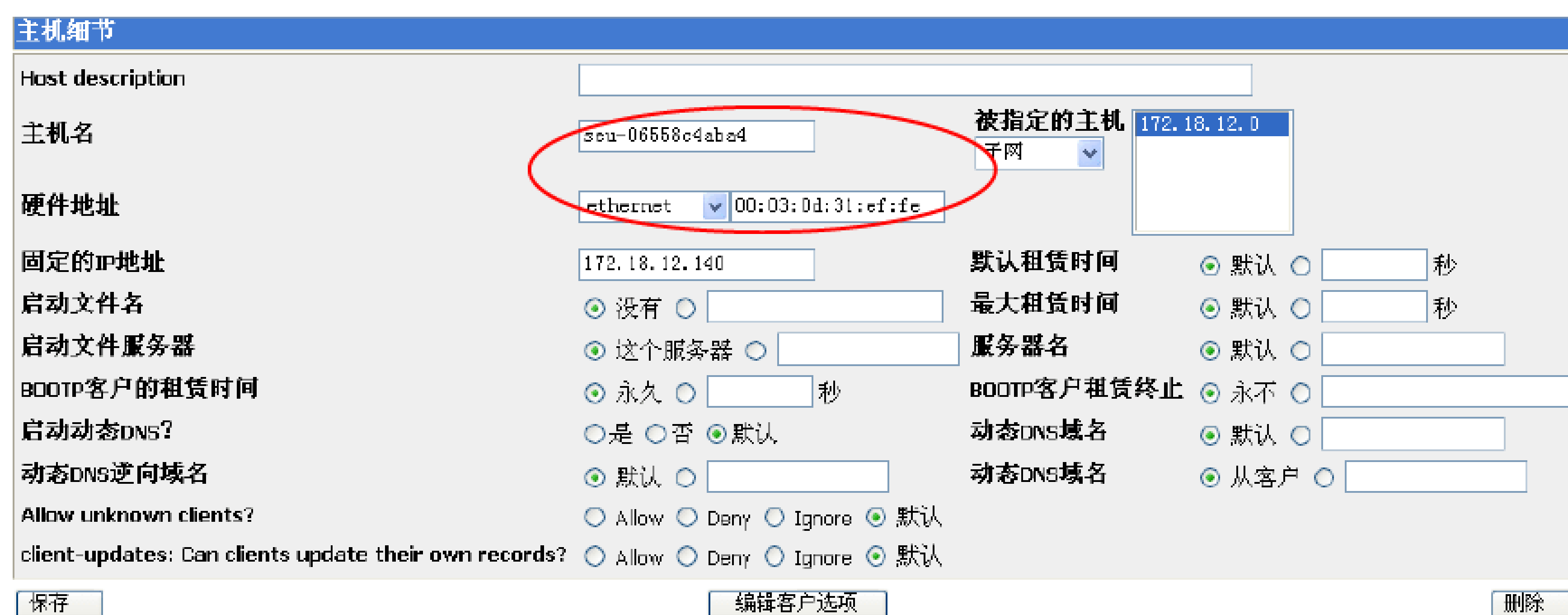


图 15-20　创建主机

❹ 由于实际情况中，公司内部常常存在多个网段的局域子网，每个子网都需要一个 DHCP 服务器为其动态分配 IP，为了节约资源，通常利用一台多网卡的主机同时作为不同的局域子网 DHCP 服务器，这就是 DHCP 共享网络存在的意义。

按照步骤(2)所演示的那样再创建一个 192.168.10.0 的子网，将 IP 范围设为 192.168.10.1～192.168.10.33, IP 的预设有效期设为 1 天，最长为 3 天。

接着，选择【增加一个新的共享网络】命令，弹出如图 15-22 所示的创建共享网络的配置窗口。【网络名】文本框设为 Two-SubNet，设置租赁时间为 1 天、3 天。【本共享网络中

的子网】文本框已经将所创建的两个子网加入其中，如图 15-22 标注部分所示。单击【新建】按钮创建共享网络，成功后出现共享网络图标，如图 15-23 所示。

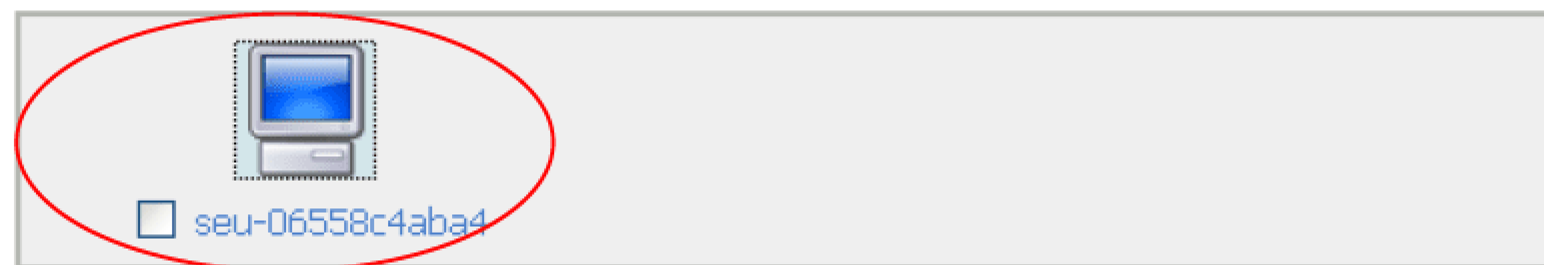


图 15-21 创建主机成功

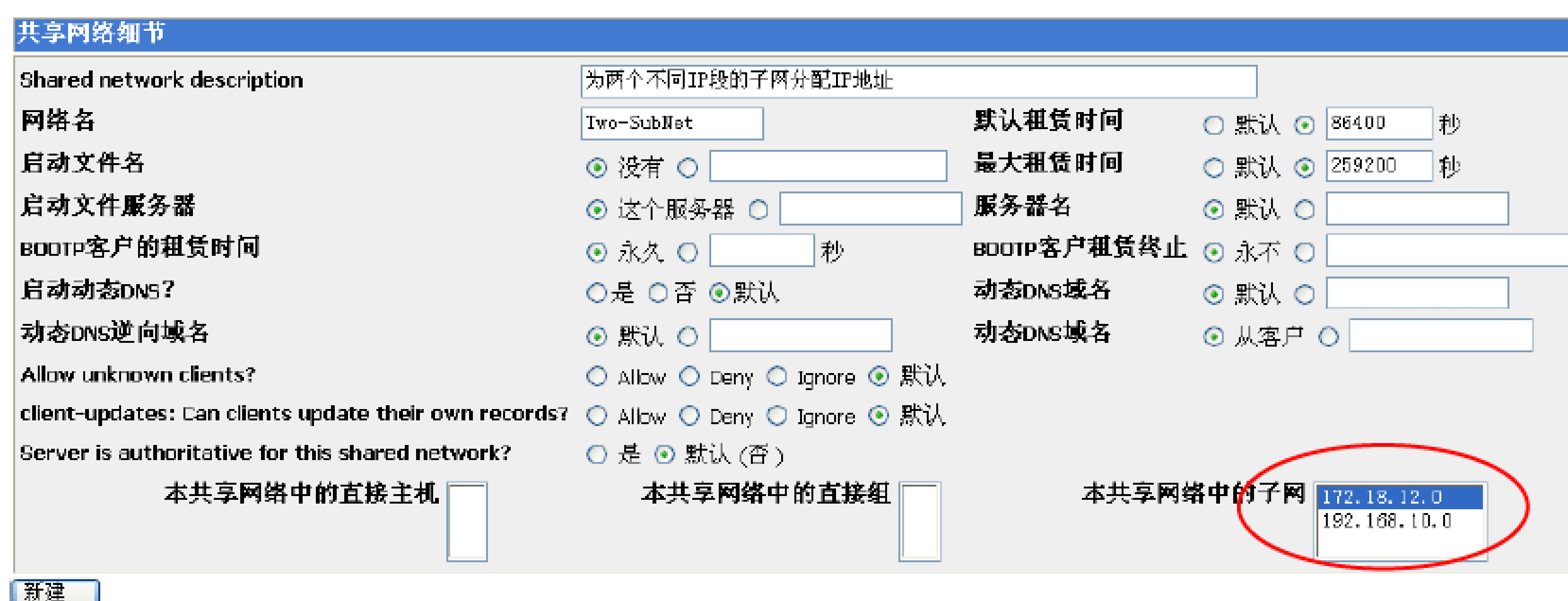


图 15-22 创建共享网络

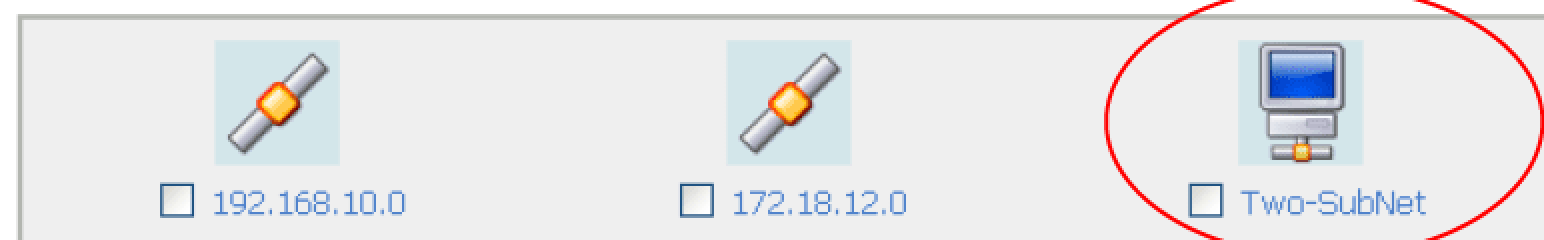


图 15-23 创建共享网络成功

📖**点评与拓展：** 作为 DHCP 共享网络的服务器必须是多网卡的，每块网卡接入一个局域子网，具有该局域网段内的 IP 地址，如拓扑图 15-16 示例的一样，Webmin 仅仅是在底层网络架设好的基础上，对 DHCP 子网和共享网络完成上层的配置工作。

15.4　使用 Webmin 配置 Samba 服务

应用实例导航——利用 Webmin 架设 Samba 服务器

※场景呈现

A 公司利用 Webmin 配置 Samba 服务器，将/home/sharing 目录开放给所有主机(包括 Linux、Windows XP)共享，Samba 服务器登录用户是系统用户 tian，并开放 tian 的根目录。

※技术要领

(1) 利用 Webmin 配置 Samba 服务器端。
(2) 利用 Webmin 设定 Samba 服务用户。

从第 13 章“Samba 服务器的配置与架设”的介绍知道，Samba 服务用于实现 Linux 服务器和 Windows 客户端的目录共享。本节主要讲述如何在 Webmin 图形窗口下设置 Samba 服务，详细配置过程如下。

❶ 打开 Webmin 管理窗口，在左侧菜单区域选择 Servers→Samba Windows File Sharing 命令，打开【Samba 共享管理器】窗口，显示出【全局配置】区域窗口，如图 15-24 所示。如果以前没有配置过 Samba 服务，则按如下步骤进行配置。

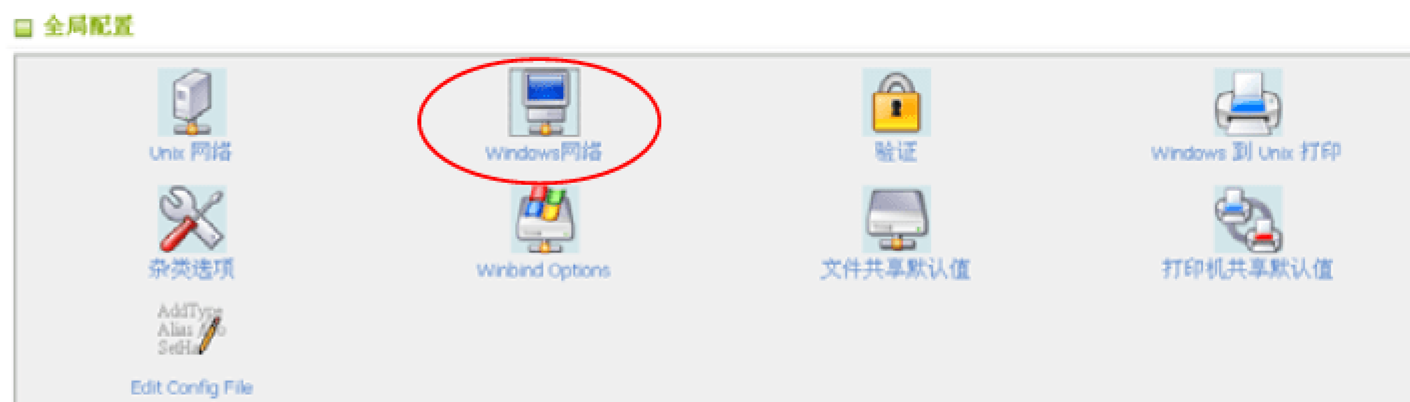


图 15-24　设置 Samba 服务窗口

❷ 单击【全局配置】中的【Windows 网络】图标，打开【Windows 网络选项】窗口，如图 15-25 所示。在【工作组】文本框中输入工作组名 MSHOME，此参数最为重要，需要与 Windows 中的工作组名称一致。如何查看 Windows 的工作组在第 13 章已经介绍过，在此不再赘述。其他参数使用默认配置即可，如在【WINS 模式】选项区选择【没有任何一个】单选按钮，在【服务器描述】文本框中输入 MSHOME Samba Server 等，最后单击【保存】按钮保存全局参数设置。

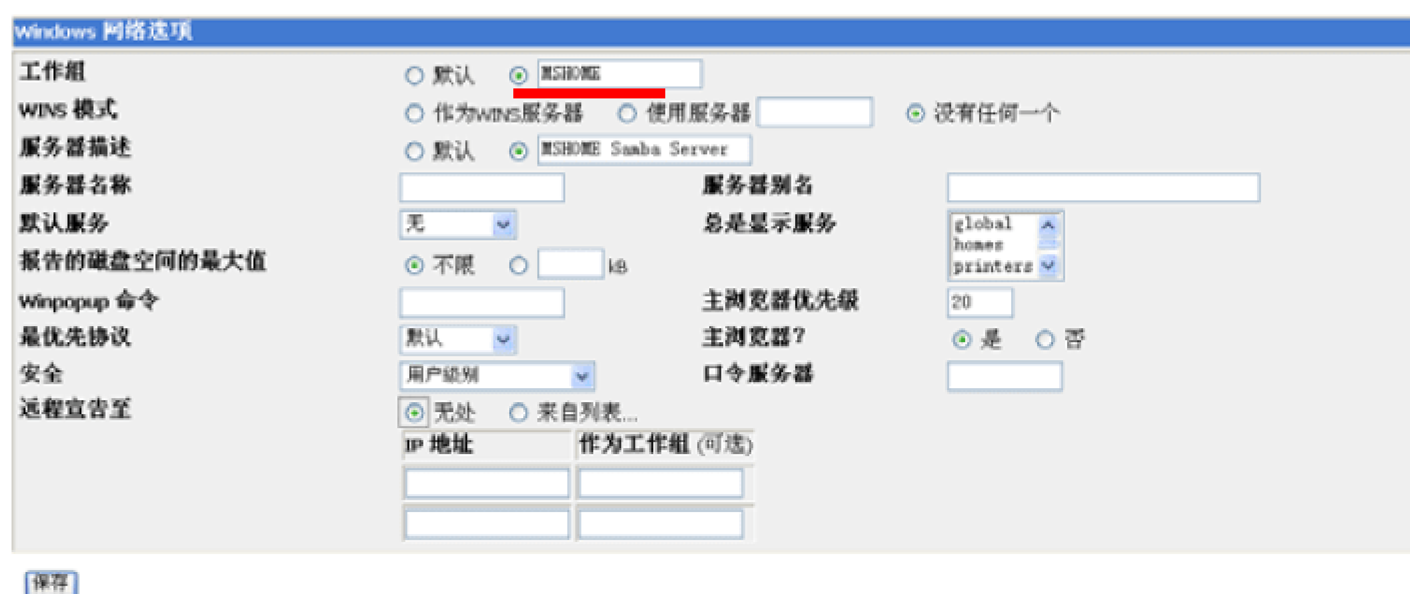


图 15-25　Windows 网络全局参数的设置

❸ 单击 Samba Users 区域中的【将 UNIX 用户转换为 Samba 用户】图标，如图 15-26 所示，进行转换用户的配置如图 15-27 所示，打开【转换用户】页面后，单击【转换用户】按钮弹出【选择多个用户】对话框，此对话框列出了 Linux 中的所有系统用户，本例我们选择用户 tian，并单击【确定】按钮，就完成了 tian 用户的转换。

若要修改 Samba 用户密码及其他信息，单击 Samba Users 区域中的【编辑 Samba 用户和口令】图标，然后选择需要编辑的 Samba 用户，此处选择 tian 用户，弹出如图 15-28 所示的【编辑 Samba 用户】窗口，可以方便地对用户 tian 的相关信息进行编辑，最后单击【保存】按钮完成编辑。



图 15-26　Samba Users 页面



图 15-27　设置转换用户



图 15-28　编辑 Samba 用户

❹ 在【Samba 共享管理器】窗口中单击上方的【创建新的文件共享】命令，如图 15-29 所示，打开【创建文件共享】窗口，如图 15-30 所示。在【共享名】区域内，选择文本框前的单选按钮。单击【共享的目录】后的【…】浏览按钮，即弹出【选择目录】对话框，在目录列表选中/home/sharing 目录，单击 OK 完成共享目录本地路径的设置。在“Automatically create directory?”选项区，选中【是】单选按钮，在 Create with owner 后的文本框内输入账户名 root，以便让程序以用户 root 的身份创建目录 sharing。其他项采用默认设置，单击【新建】按钮创建新的共享目录。返回如图 15-31 所示的共享列表，在共享目录列表中可以看到新创建的共享目录/home/sharing 以 public 的方式进行共享。

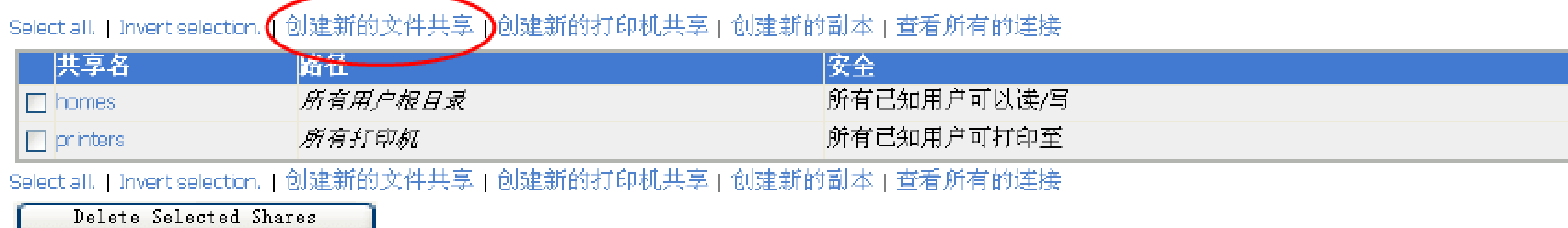


图 15-29　Samba 共享管理服务窗口

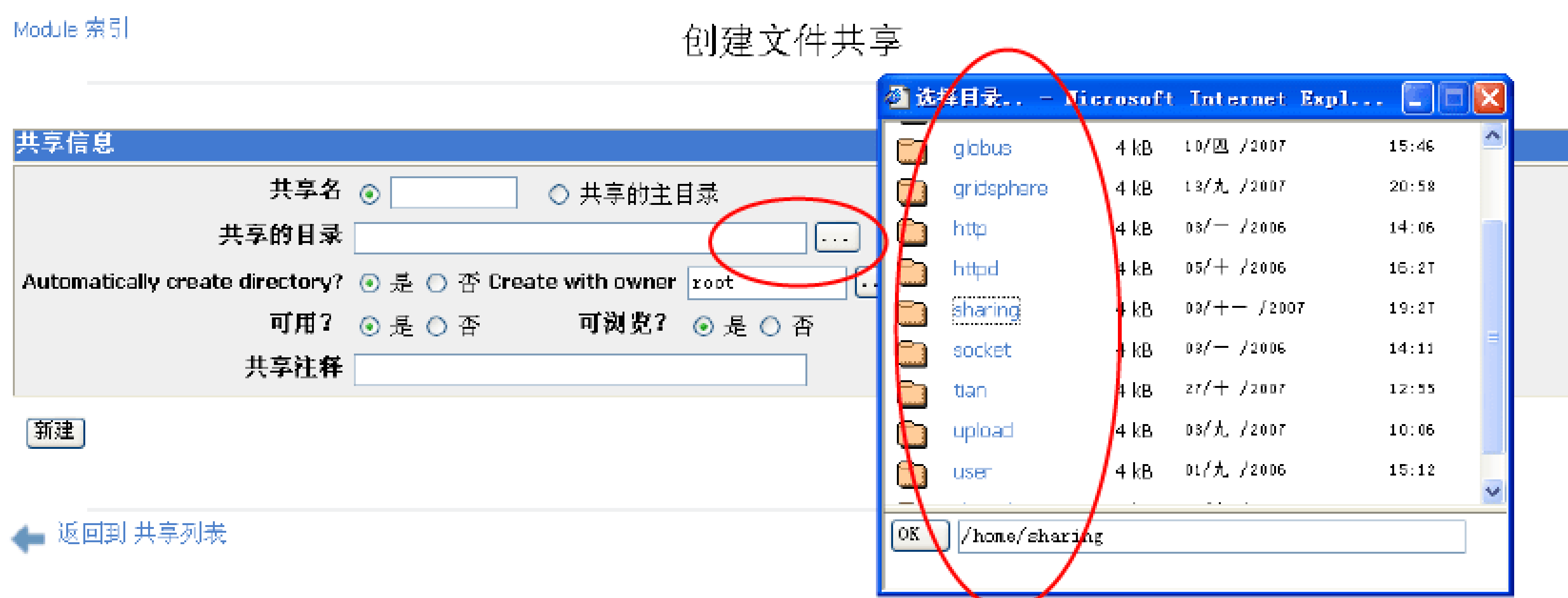


图 15-30　创建共享目录

Select all. | Invert selection. | 创建新的文件共享 | 创建新的打印机共享 | 创建新的副本 | 查看所有的连接

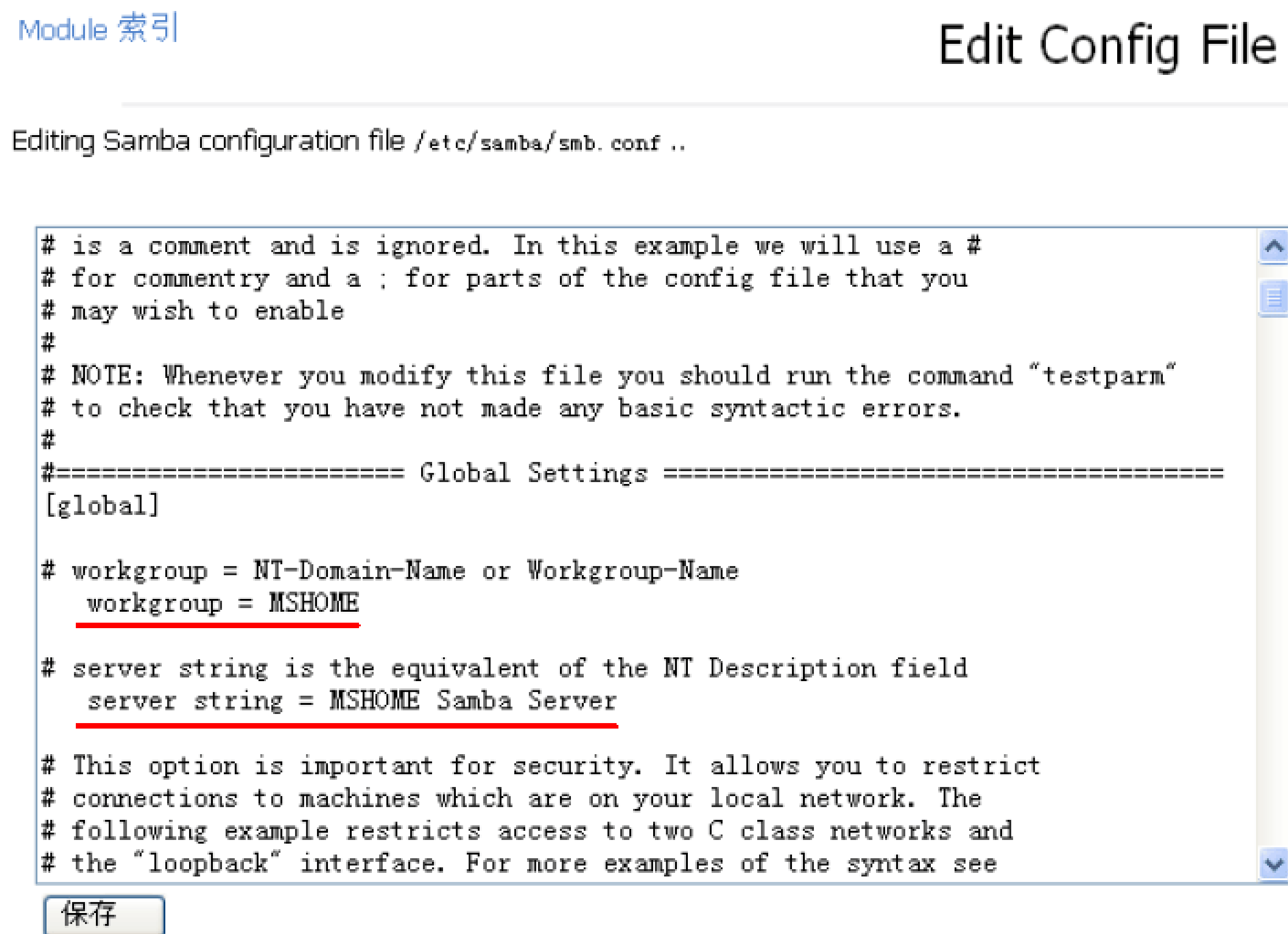
共享名	路径	安全
homes	*所有用户根目录*	所有已知用户只读
printers	*所有打印机*	所有已知用户可打印至
public	/home/sharing	都可以读/写

Select all. | Invert selection. | 创建新的文件共享 | 创建新的打印机共享 | 创建新的副本 | 查看所有的连接

Delete Selected Shares

图 15-31　新建文件共享后的共享目录列表

❺ 单击图 15-24 所示【全局配置】区域内的 Edit Config File 图标，打开编辑配置文件窗口，如图 15-32 所示。在此窗口下，用户可以编辑修改 Samba 服务的全局参数。如设置 Samba 服务器所属群组名称，使用 workgroup=MSHOME 语句；设置 Samba 服务器简要说明，可以使用 server string=MSHOME Samba Server 语句。


Module 索引

Edit Config File

Editing Samba configuration file /etc/samba/smb.conf ..

```
# is a comment and is ignored. In this example we will use a #
# for commentry and a ; for parts of the config file that you
# may wish to enable
#
# NOTE: Whenever you modify this file you should run the command "testparm"
# to check that you have not made any basic syntactic errors.
#
#======================= Global Settings =====================================
[global]

# workgroup = NT-Domain-Name or Workgroup-Name
  workgroup = MSHOME

# server string is the equivalent of the NT Description field
  server string = MSHOME Samba Server

# This option is important for security. It allows you to restrict
# connections to machines which are on your local network. The
# following example restricts access to two C class networks and
# the "loopback" interface. For more examples of the syntax see
```

保存

图 15-32　编辑 Samba 配置文件

15.5　使用 Webmin 配置 DNS 服务

应用实例导航——利用 Webmin 架设 DNS 服务器

※场景呈现

A 公司利用 Webmin 配置管理公司域的 DNS 服务器，所管理的域名为 seu.edu.cn。该

DNS 服务器需要提供该域内的所有主机的正向解析；该域的 IP 范围为 172.18.12.*，该 DNS 服务器也需要提供对这个范围内的 IP 地址的反向解析功能，并开放 IP 地址为 211.65.63.146 的服务器为从 DNS 服务器。

※技术要领

(1) 利用 Webmin 配置 DNS 服务器端。
(2) 利用 Webmin 创建 DNS 正反解析数据库文件。
(3) 利用 Webmin 开放从 DNS 服务器。

从第 7 章“DNS 服务器的配置与架设”的介绍知道，对 DNS 的配置包括对主配置文件/etc/named.conf 的设定、正反解域数据库文件的创建等，本节主要讲述如何在 Webmin 图形窗口下配置 DNS 服务器，其步骤和原理与第 7 章中 shell 下的配置一样。

本节主要讲述如何在图形管理工具 Webmin 下面设置 DNS 服务。本节中的例子以前面相关章节的部分实例为基础，具体参数设置的作用可以参见第 7 章的介绍。

❶ 打开 Webmin 管理窗口，在左侧菜单区域选择 Servers→BIND DNS Serve 命令，打开【BIND DNS 服务器】管理窗口，如图 15-33 所示。



图 15-33　BIND DNS 服务器设置窗口

❷ 单击【现有 DNS 区域】选项区内的【创建新的主区域】图标，打开 【创建主区域】窗口，如图 15-34 所示，该窗口用于创建正解区域和反解区域。
若要创建正解区域，则在【区域类型】选项区中选择【正向(名称至地址)】单选按钮。在【域名/网络】后的文本框中输入要创建的主域名 seu.edu.cn。在【记录文件】选项区中选择【自动】单选按钮，单击【…】按钮，即弹出【选择文件】对话框，选择 named.seu.edu.cn 文件名，即选择正解文件的路径，然后单击 OK 按钮保存记录文件名。在【主服务器】文本框中输入服务器主机名 sec 或其 IP 地址，并需要设置刷新时间和过期时间，最后单击【新建】按钮创建新的正解区域。

创建反解区域的方法与创建正解区域基本相同，在【区域类型】选项区中选择【逆向(地址至名称)】单选按钮，然后在【域名/网络】文本框中输入要创建的反向域名称 172.18.12;在【记录文件】选项区中选择【自动】后的单选按钮，定位到反向解析数据库文件named.172.18.12。其他设置与创建主区域的方法相同。

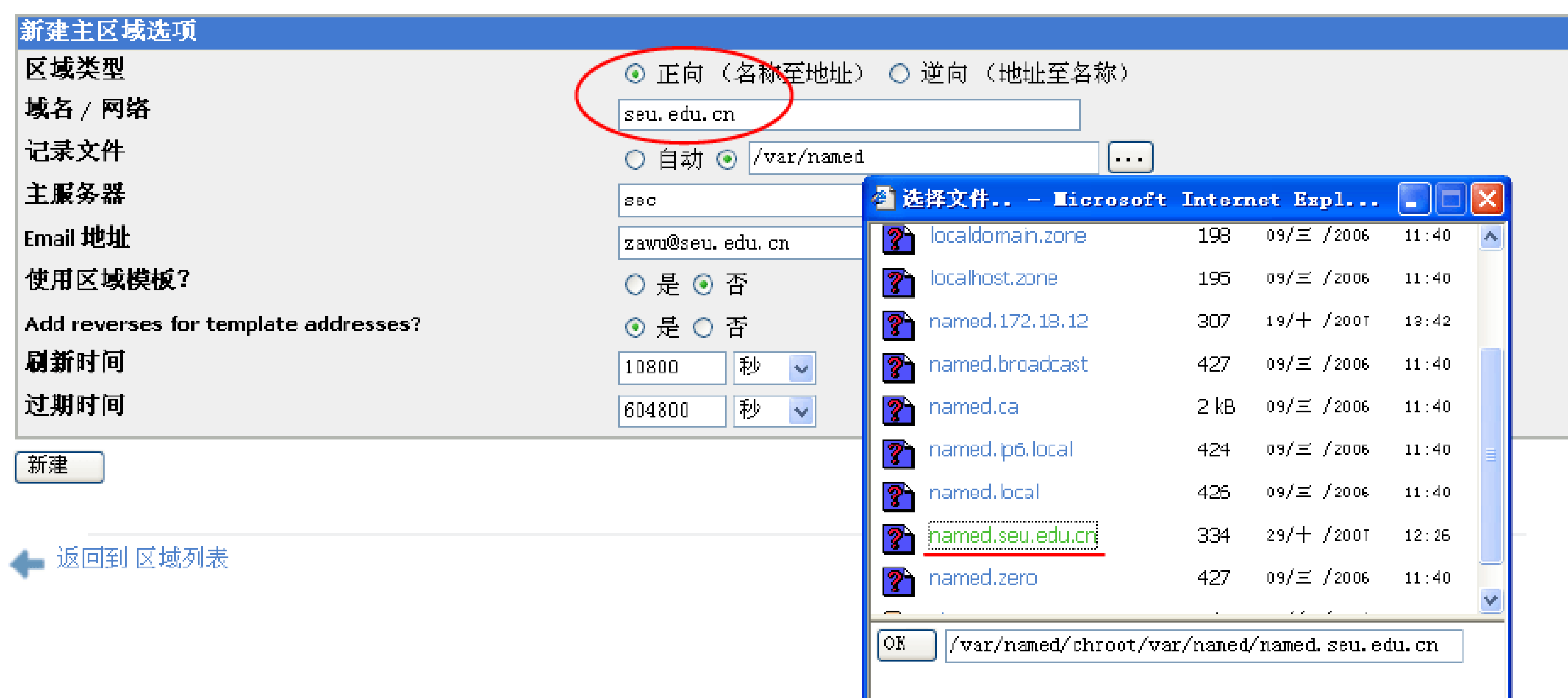


图 15-34　创建主区域

分别创建以上的正解区域和反解区域后，返回到如图 15-35 所示的区域列表，则会看到新创建的 172.18.12 和 seu.edu.cn 区域文件。

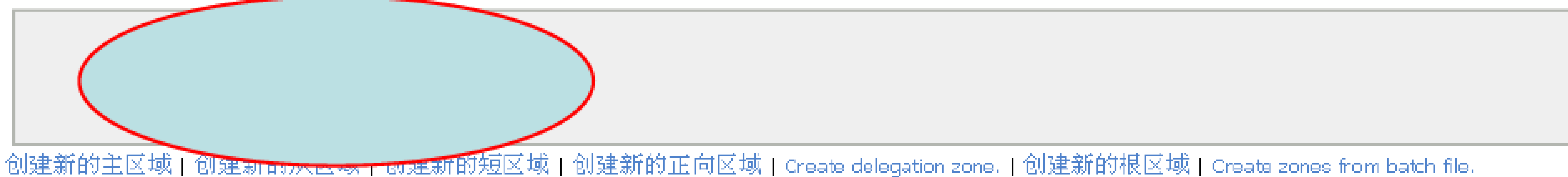


图 15-35　完成新建后的区域列表

❸ 单击 seu.edu.cn 图标，打开【编辑主区域】窗口，如图 15-36 所示。单击【地址】图标，打开【地址个记录】窗口，如图 15-37 所示，在此可以编辑主机地址名称和 IP 地址的映射表。【编辑主区域】窗口中的【命名服务器】提供了设置从 DNS 服务器的功能。单击【命名服务器】图标，打开如图 15-38 所示的【命名服务器个记录】窗口，在此可以添加和删除从 DNS 服务器地址。



图 15-36　DNS 编辑主区域窗口

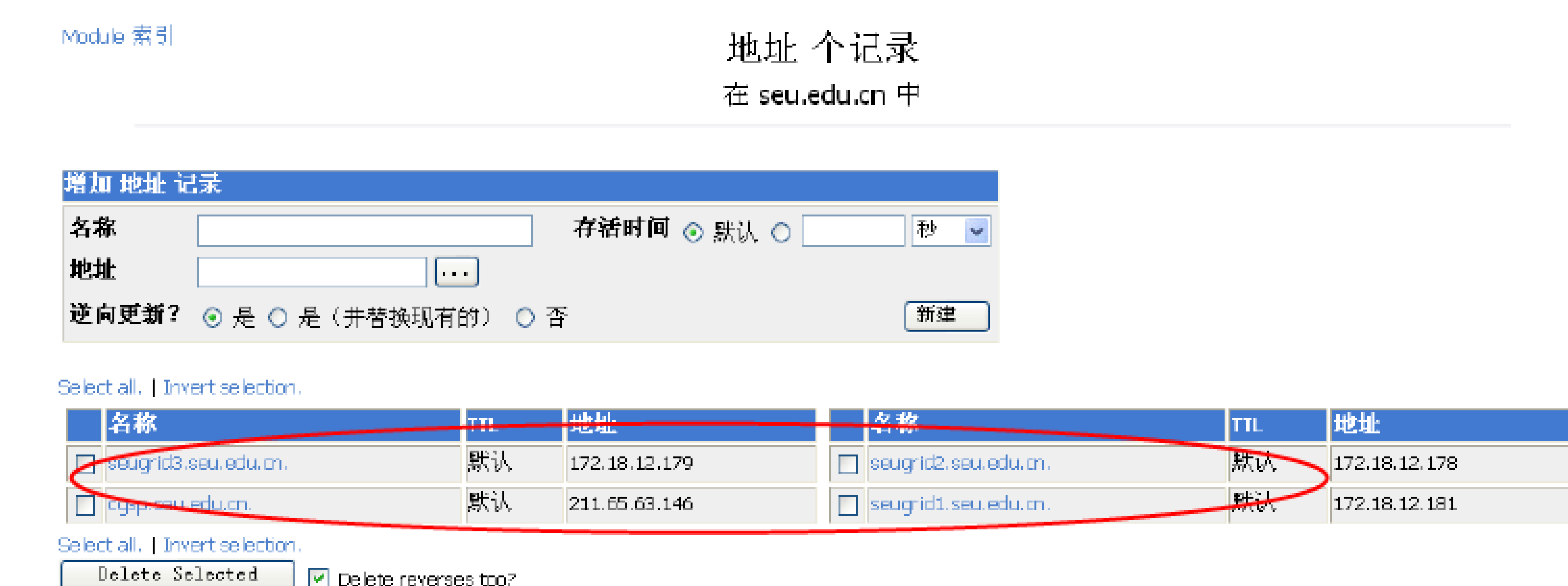


图 15-37　主机地址名称和 IP 地址的映射表

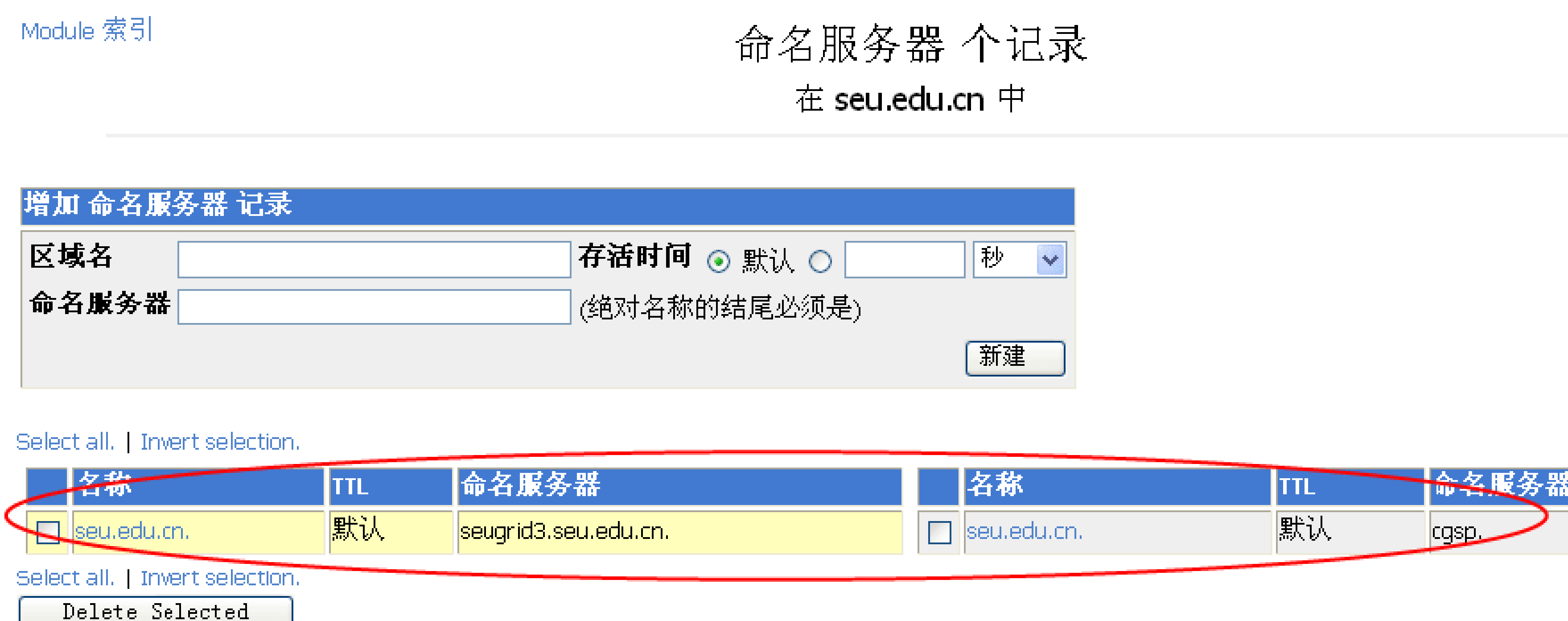


图 15-38　编辑从 DNS 服务器记录

❹【编辑主区域】窗口下方的菜单提供了对所创建的 DNS 区域的编辑功能，菜单内容如图 15-39 所示。单击【编辑记录文件】图标，出现如图 15-40 所示的【编辑记录文件】窗口，窗口上方显示的/var/named/named.seu.edu.cn 实际上是正向解析数据库文件在 chroot 中的路径，这

也正是 chroot 的作用所在。从第 6 章的介绍中，我们知道 named.seu.edu.cn 的真实路径是 /var/named/chroot/var/named。利用【编辑记录文件】窗口就可以方便地更改正向解析数据库文件的内容。

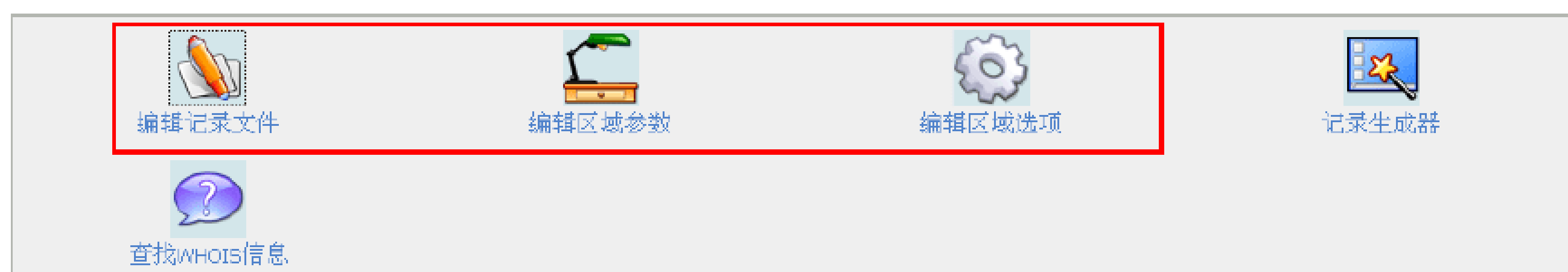


图 15-39　编辑主区域设置窗口

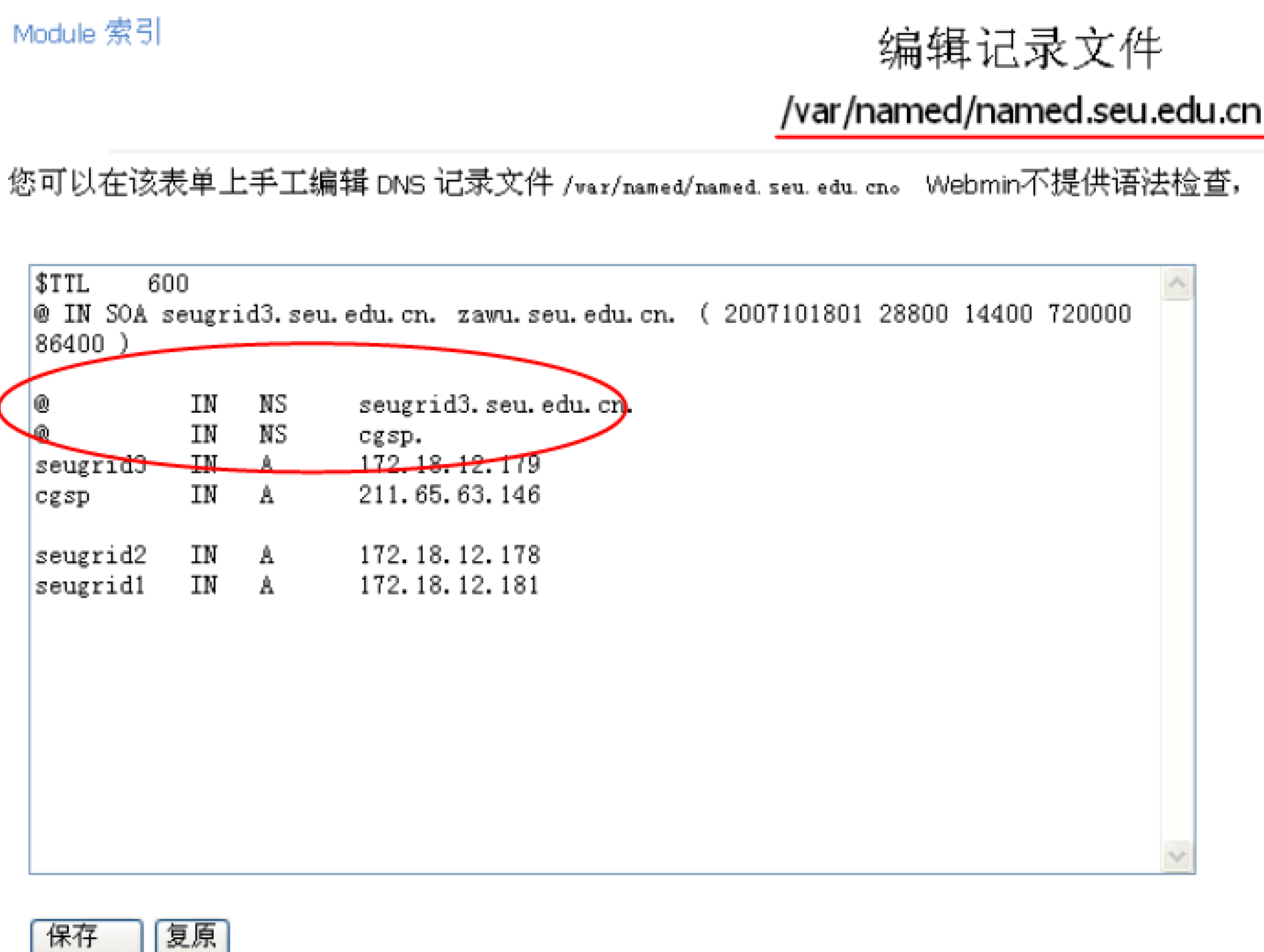


图 15-40　编辑 var/named/named.seu.edu.cn 文件

若单击【编辑区域参数】图标，则将打开【区域参数】窗口，如图 15-41 所示，在这里可以修改主服务器名称，管理员 E-mail 地址，以及其他诸如 TTL 等 SOA 资源记录的选项参数，最后单击【保存】按钮保存所有的区域参数设置。【编辑主区域】窗口下方的【编辑区域选项】图标用于开放从 DNS 服务器，单击后出现如图 15-42 所示的【区域选项】窗口，其中【允许传输来自…】文本框里输入从 DNS 服务器地址，本例为 211.65.63.146，表示开放与 DNS 从服务器之间的传输文件的权限，单击【保存】按钮保存设置。

5 在图 15-33 中【全局服务器选项】区域内，单击【转发和传输】图标，打开 【全域转发和区域传输选项】窗口，如图 15-43 所示。在【转发查询到服务器】文本框的 IP address 列中输入需要转发到的 DNS 服务器地址 202.119.24.12 和 202.119.24.18。其他选项设置为【默认】，单击【保存】按钮保存设置。设置完成后就实现了这样的功能：缓存服务器首先将查询转发给第一台 DNS 服务器，如果第一台无应答，则将查询转发给第二台 DNS 服务器，依次类推，直到接收到来自 DNS 服务器的确定应答。

Module 索引

区域参数
seu.edu.cn

区域参数			
主服务器	seugrid3.seu.edu.cn.	Email 地址	zavu@seu.edu.cn
刷新时间	28800 秒	传输重试时间	14400 秒
过期时间	720000 秒	默认的活动时间	86400 秒
Default time-to-live for records	○ 默认 ⊙ 600 秒		

保存

图 15-41　编辑主 DNS 服务器的区域参数

Module 索引

区域选项
seu.edu.cn

区域选项			
是否检查名称?	○ 警告 ○ 失败 ○ 忽略 ⊙ 默认	是否通知从区域的更改?	○ 是 ○ 否 ⊙ 默认
允许更新自...		允许传输来自...	211.65.63.146
允许查询自...		也通知从区域...	

保存　　转换为从区域

图 15-42　开放从 DNS 服务器

全域转发和区域传输选项		
转发查询到服务器	IP address	Port (optional)
	202.119.24.12	
	202.119.24.18	
若转发驱动程序不响应则直接查找	○ 是 ○ 否 ⊙ 默认	
最大区域传输时间	⊙ 默认 ○ 分钟	
区域传输格式	○ 每次一个 ○ 多个 ⊙ 默认	
最大并发区域传输数	⊙ 默认 ○	

保存

图 15-43　【全域转发和区域传输选项】窗口

❻ 在图 15-33 所示的【BIND DNS 服务器】管理窗口中的【现有 DNS 区域】区域内，单击【创建新的从区域】图标，打开【创建从区域】窗口，如图 15-44 所示。在【区域类型】选项区中选择【正向解析(名称至地址)】单选按钮。在【区域名/网络】文本框中输入区域名 seu.edu.cn.second。在【记录文件】选项区中选择【自动】后的单选按钮，单击【…】按钮选择记录文件路径/var/named/chroot/var/named/named.172.18.12。在【主服务器】文本框中输

入该域名所在的 DNS 服务器 IP 地址 172.18.12.179。最后单击【新建】按钮创建新的从 DNS 服务器。创建反向解析从 DNS 服务器的方法与正向解析的方法基本相同，在此不再赘述。

Module 索引

创建从区域

新建从区域选项

区域类型 ⊙ 正向解析（名称至地址） ○ 逆向解析（地址至名称）

区域名 / 网络 seu.edu.cn.second

记录文件 ○ 无 ○ 自动 ⊙ /var/named/chroot/var/named/nam ...

主服务器 172.18.12.179 服务器端口 ⊙ 默认 ○ 端口

新建

图 15-44　创建从 DNS 服务器

15.6　使用 Webmin 配置 Web 服务

应用实例导航——利用 Webmin 架设 Web 服务器

※场景呈现

A 公司利用 Webmin 配置 Apache Web 服务器，将 Web 服务器的根目录设为/var/www/html，并创建几个虚拟目录，设定其中的一台虚拟主机，IP 为 192.168.10.11，目录为/usr/www/web。

※技术要领

(1) 利用 Webmin 设置 Apache Web 服务器的根目录。
(2) 利用 Webmin 创建 Apache Web 虚拟主机。

本节讲述如何在图形管理工具 Webmin 下面配置 Apache Web 服务器，并实现虚拟主机功能，具体步骤如下。

❶ 打开 Webmin 管理窗口，在左侧菜单区域选择 Servers→Apache Webserver 命令，打开【Apache WEB 服务器】管理窗口，如图 15-45 所示。在此主窗口下对 Apache 服务器进行配置。

❷ 首先，介绍如何在 Webmin 下设置根目录，单击【默认服务器】图标，如图 15-45 所示，打开【虚拟服务器选项】窗口，如图 15-46 所示。单击【文档选项】图标，打开【文档选项】窗口，如图 15-47 所示。在【文档根目录】选项区中选择【默认】后的单选按钮，单击【…】按钮选择新目录路径/var/www/html。 在【用户 www 目录】部分设置目录路径的访问权限，这里选择【默认】后的单选按钮，在其后的文本框中输入 disable，然后再选中【所有用户均可访问】前的单选按钮。在【按目录配置的选项文件】选项区中选择【默认】后的单选按钮

钮，在其后的文本框中输入.htaccess。单击【保存】按钮保存设置。



图 15-45　Apache Web 服务器设置窗口



图 15-46　设置默认服务器选项

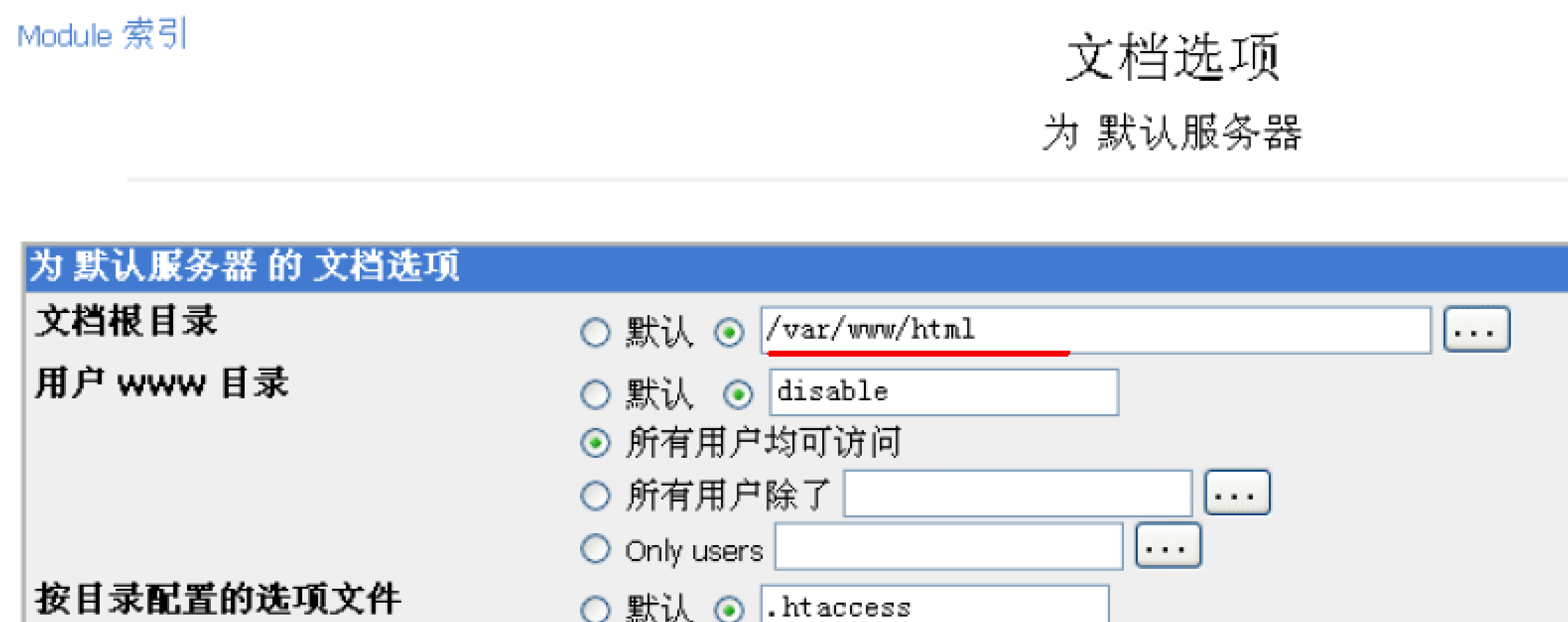


图 15-47　设置默认服务器主目录路径

❸ 对根目录设置完毕后，需要设置 Web 服务器的主机名。单击【网络和地址】图标，打开【网

络和地址】设置窗口，如图 15-48 所示。在【服务器管理员电子邮件地址】区域选择【无】后的单选按钮，在其后的文本框中输入管理员的 E-mail 地址。在【服务器主机名称】区域选择【自动】后的单选按钮，在其后的文本框中输入服务器域名 seugrid3.seu.edu.cn，如果没有域名则输入服务器 IP 地址。单击【保存】按钮，保存设置。



图 15-48　设置管理员 E-mail 地址和服务器名称

❹ 除了根目录以外，Apache Web 服务器还灵活支持虚拟目录，所谓虚拟目录，是指 Web 服务器上显示目录的路径并非实际的物理路径，而是通过映射表寻找到物理路径进行定位。那么如何设定 Web 服务器的虚拟目录呢？单击【别名和重定向】图标，打开【别名和重定向】窗口，如图 15-49 所示。在【文档目录别名】文本框的列表中输入要设置的虚拟目录。在【从】一列中输入虚拟目录路径，在【到】一列中输入对应的物理路径。图中分别创建了 3 个虚拟目录：名为/icons 的虚拟目录及其对应物理路径/var/www/icons；名为/error 的虚拟目录及其对应物理路径/var/www/error；名为/usage 的虚拟目录及其对应物理路径/var/www/usag。单击【保存】按钮保存设置。



图 15-49　设置虚拟目录

❺ 在安全控制方面，Apache Web 服务器为根目录和虚拟目录提供了全面的访问控制设定，我们以根目录为例来说明。单击【默认服务器】图标，打开如图 15-50 所示的窗口。单击【按目录选项】区域内要设置权限的目录 Directory/var/www/html，打开【按目录选项】窗口，如图 15-51 所示。在【按目录选项】区域内，单击【访问控制】图标，打开【访问控制】设置窗口，在【限制访问】区域按图 15-52 所示进行各种相关访问控制参数的设置，可以对 IP 地址、子网段及主机名进行允许或拒绝访问设置。最后单击【保存】按钮保存设置。

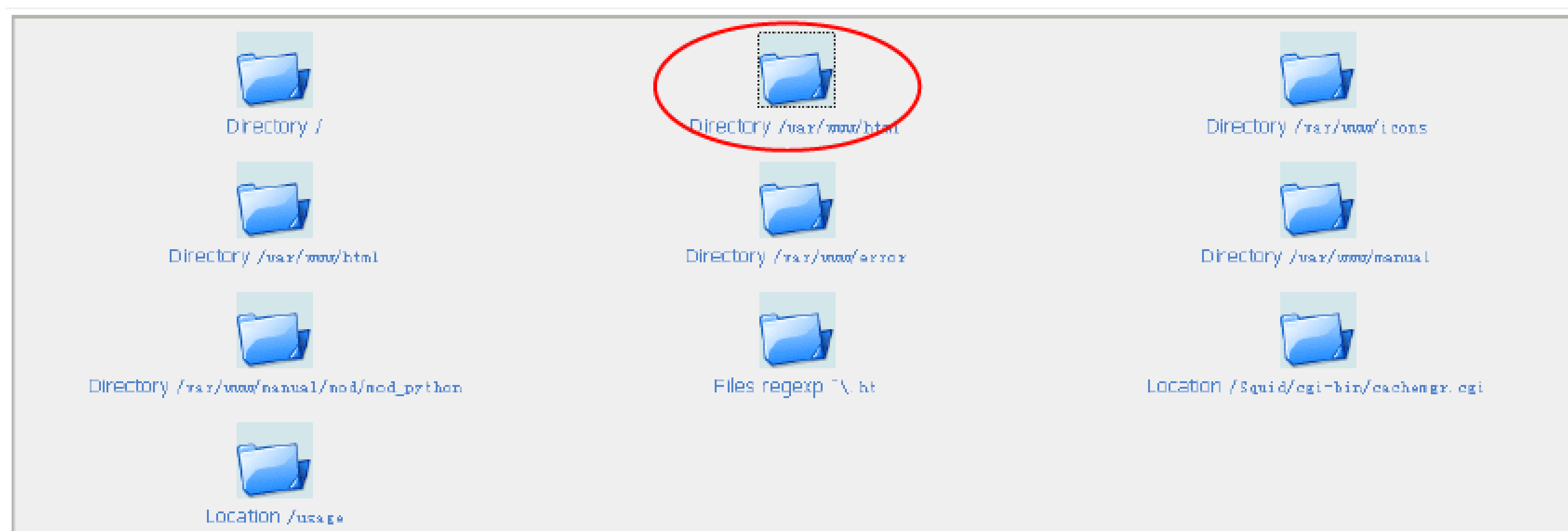


图 15-50　设置默认服务器目录选项



图 15-51　设置指定目录选项

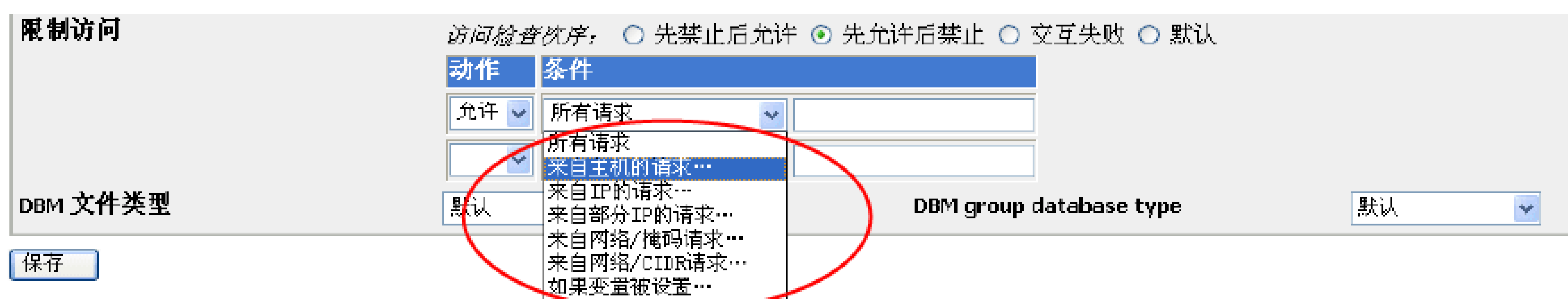


图 15-52　设置指定目录访问控制

❻ 场景要求中需要创建一个 IP 为 192.168.10.11 的虚拟主机，所谓虚拟主机是指使用一台物理机器，充当多个 IP 地址(或主机名)的服务器，例如，本例中由 Webmin 控制的这台主机的真实 IP 为 172.18.12.179，但是它通过虚拟主机的设定，又可以充当 IP 为 192.168.10.11 的 Web 服务器。使用虚拟主机的好处在于，一些小规模的网站，通过跟其他网站共享同一台物理机器，可以减少系统的运行成本，并且可以减少管理的难度。

下面介绍如何设定虚拟主机。单击图 15-45 中【Apache WEB 服务器】管理窗口下的 Create virtual host 标签，打开【创建新的虚拟服务器】窗口，如图 15-53 所示。在【处理到地址的连接】选项区中选择【指定地址…】单选按钮，在其后的文本框中输入虚拟主机 IP 地址 192.168.10.11。在【端口】选项区中选择【默认】单选按钮。在【根文件】文本框中输入虚拟主机主目录/usr/www/web。在【服务器名称】选项区中选择【自动】单选按钮。单击 Create Now

按钮就可以创建新的虚拟主机。

需要注意的是，每个不同的虚拟主机对应不同的虚拟主目录，其中可以存放架设网站的一些必需文件，网络管理员当然仍然可以按照步骤(5)中的介绍进行访问控制。



图 15-53　创建虚拟主机

15.7　使用 Webmin 配置 NFS 服务

本节讲述利用 Webmin 配置 NFS 服务，详细配置过程如下。

❶ 在 Webmin 管理窗口中，单击目录区中 Networking→NFS Exports 选项，弹出如图 15-54 所示的 NFS 配置窗口。第 5 章“NFS 和 NIS 服务器的配置与应用”曾经介绍过 NFS 主配置文件是/etc/exports，其中设定了共享目录的路径、属性、访问权限等，Webmin 的 NFS 管理首页列出了该服务器所开放的共享目录及其权限，并且提供了对所开放的共享目录激活和关闭的功能。比如我们选中图 15-54 中的/home/upload 目录，单击 Disable Selected 按钮可以将其设置为“不活”，即不对外开放，单击 Enable Selected 按钮则可以将不活动的目录再次激活，“不活”的共享目录如图 15-55 所示。



图 15-54　NFS 配置窗口



图 15-55　将共享目录设为“不活”

❷ 在 NFS 配置窗口中，单击【增加一个新的输出】选项，弹出如图 15-56 所示的新建共享目录窗口，在【要输出的目录】对话框输入所要共享目录的路径，【激活？】选项区选择【是】单选按钮。在【IPv4 网络】文本框中可以对访问共享目录的客户端进行权限设置，比如，我们需要将客户端限制在内部局域网内，则输入 172.18.12.0，在【网络掩码】文本框输入 255.255.255.0，这就说明只有 IP 为 172.18.12.*的 v4 地址才能访问该共享目录。



图 15-56　创建新的共享目录

15.8　使用 Webmin 配置 SSH 服务

本节讲述如何利用 Webmin 配置 SSH 服务，由于 sshd 进程默认安装，因此可以直接对 SSH 服务进行配置。

❶ 在 Webmin 管理窗口中，选择目录区中 Servers→SSH Server 选项，弹出如图 15-57 所示的【SSH 服务器】管理窗口。可以看到 SSH 管理器中有【网络】图标，单击此图标弹出如图 15-58 所示的【网络】设置窗口，Listen on addresses 设定项用于限制连接该 SSH 服务器的客户端地址，SSH 选择 v2 版本，其他选择默认，最后单击【保存】按钮保存设置。



图 15-57　SSH 服务器设置窗口

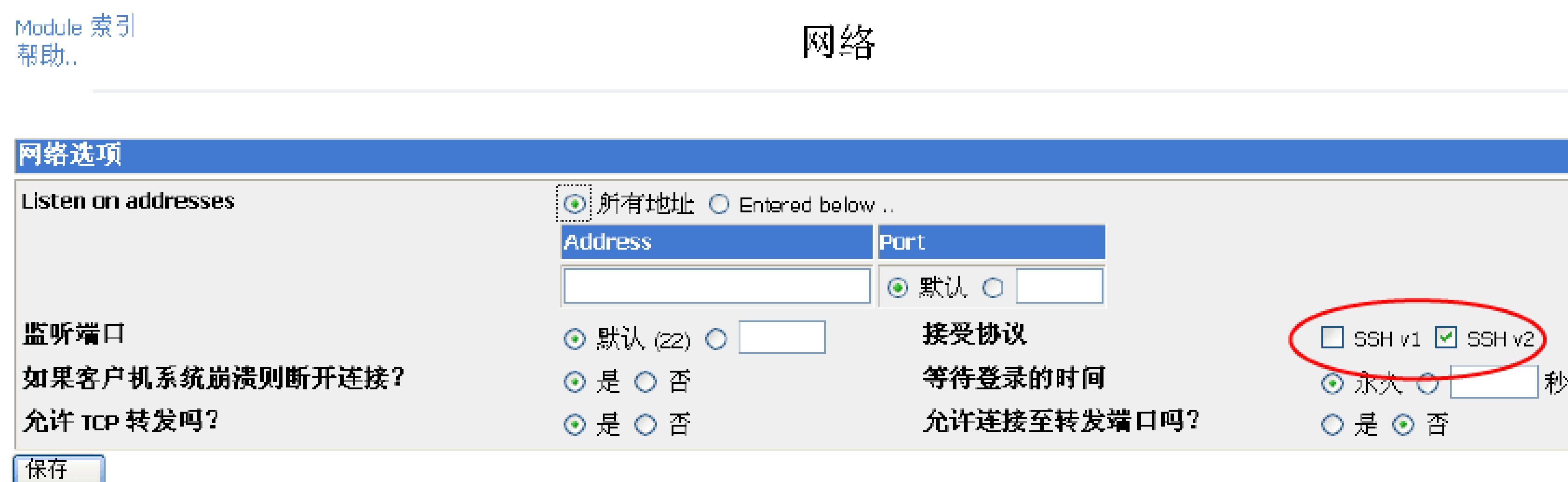


图 15-58　SSH 网络设置

❷ SSH 服务器的管理窗口中的 Edit Config Files 图标用于编辑 SSH 服务的主配置文件，即/etc/ssh/sshd_config 文件，如图 15-59 所示，利用它可提供可视化的文本编辑窗口。

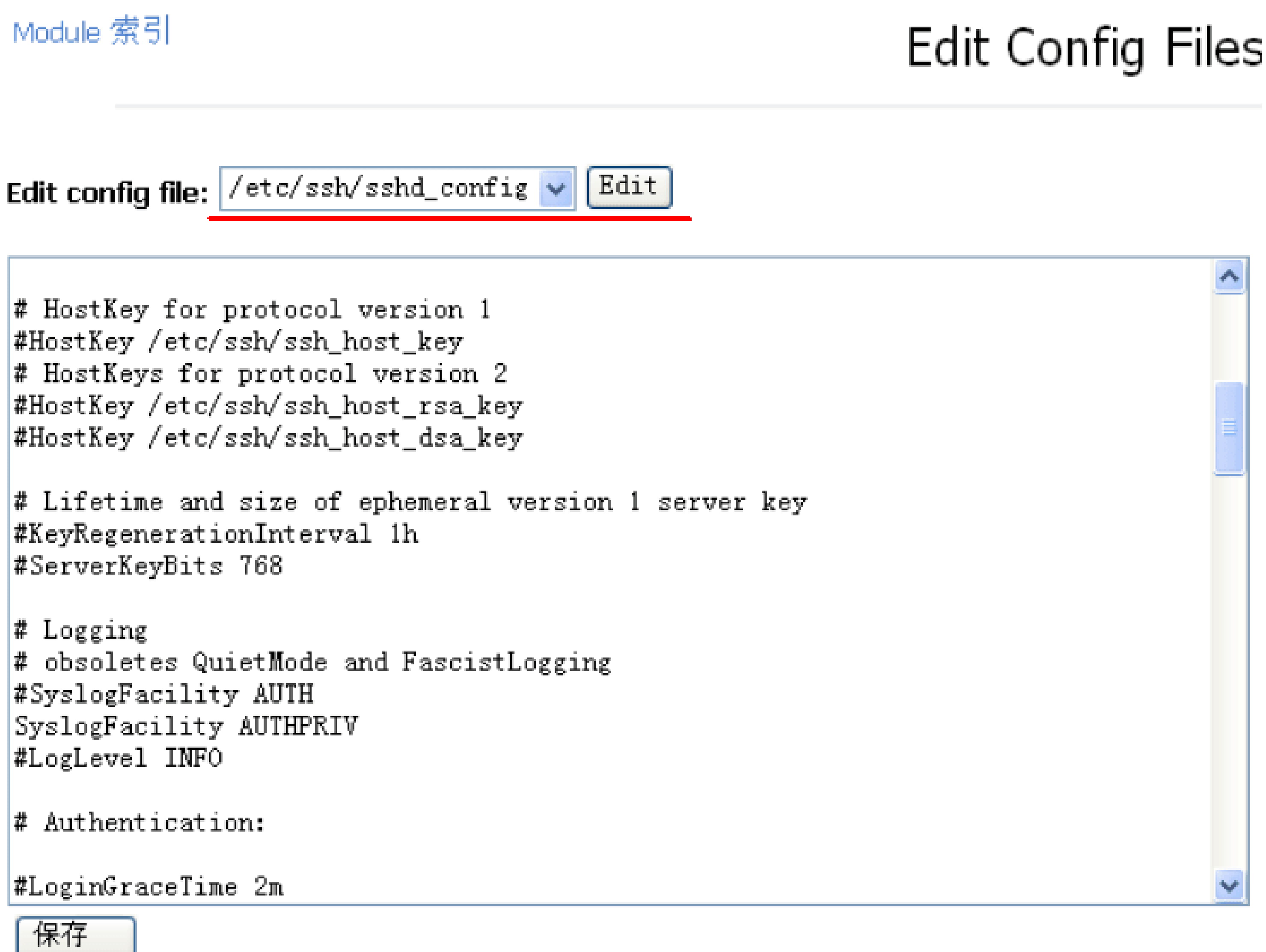


图 15-59　编辑 SSH 配置文件

15.9　使用 Webmin 配置防火墙服务

应用实例导航——利用 Webmin 架设 Linux 防火墙

※场景呈现

A 公司利用 Webmin 进行架设 Linux 防火墙，通过添加 iptables 的 filter 表的规则实现默认策略的制定，实现对 TCP、UDP 数据包各种条件下(如源、目标 IP 地址，源、目标端口号等)的过滤功能，并对某些类型的 ICMP 数据包进行控制。并通过对 NAT 表的配置，将所有访问网关 80 端口的 IP 包转发局域网 Web 服务器的 8080 端口。

※技术要领

(1) 理解 Linux 防火墙组件 iptables 的原理和结构。
(2) 利用 Webmin 为 filer 表和 nat 表的各条链添加规则。

从第 3 章“Linux 防火墙与 NAT 服务”的论述中，我们知道 Linux 防火墙通过 iptables 内置的 3 张表——filter 表、nat 表和 mangle 表，来实现包过滤、网络地址转换和包重构等功能。Webmin 提供了对这 3 张表的管理功能，下面讲述如何通过 Webmin 管理和配置防火墙。

❶ 在 Webmin 管理窗口，选择目录区中 Networking→Linux Firewall 选项，弹出如图 15-60 所示的 Linux 防火墙管理窗口，该窗口默认显示 filter 表三条链中的规则，可以通过管理窗口上方的 Showing IPtables 按钮切换 iptables 表。我们可以注意到，filter 链的表格有四个表头，分别为 Action、Condition、Move 和 Add，其中 Action 显示了对报文是接受还是丢弃，Condition 规定了对何种条件的报文做 Action 中规定的处理，Move 按钮可以改变链中规则的顺序，Add 按钮用于添加新规则。

❷ 比如添加禁止客户机访问聊天网站的规则，只要在如图 15-61 所示 FORWARD 链表格中，单击 Add Rule 按钮，弹出添加规则窗口。其中，Rule Comment 文本框记录该规则的意思，便于用户理解，无实际用处；首先在 Action to take 选项区中选择 Drop 单选按钮，表示该规则是将符合条件的 IP 包丢弃，如图 15-62 所示；然后设置规则的条件，在 Destination address or network 标签后的下拉列表框中选择 Equals，后面的文本框输入所要禁止的聊天网站的地址或 IP，如图 15-63 所示。然后单击【新建】按钮创建规则，这条规则的意思是：所有发送到聊天网站的 IP 包全部丢弃，从而达到禁止客户机访问聊天网站的目的。



图 15-60　Webmin 提供的防火墙管理窗口

Forwarded packets (FORWARD)

Select all. | Invert selection.

Action	Condition	Move	Add
Drop	If destination is 209.62.21.206		
Drop	If destination is 209.62.21.206		
Drop	If destination is 209.62.21.206		
Drop	If destination is 202.17.61.4		
Drop	If destination is 216.163.137.3		

Select all. | Invert selection.

Set Default Action To: Accept　Delete Selected　Add Rule

图 15-61　FORWARD 链的表格

Module 索引

Add Rule

Chain and action details	
Part of chain	Incoming packets (INPUT)
Rule comment	deny chatting room
Action to take	○ Do nothing ○ Accept ⊙ Drop ○ Reject ○ Userspace ○ Exit chain ○ Log packet ○ Run chain
Reject with ICMP type	⊙ 默认 ○ Type icmp-net-unreachable

图 15-62　为规则添加动作

Condition details		
Source address or network	<Ignored>	
Destination address or network	Equals	www.chattingroom.com

图 15-63　为规则添加条件

❸ 再比如添加禁止子网客户机使用 FTP 的规则，还是在 FORWARD 链表格中，单击 Add Rule

按钮，弹出添加规则窗口，在 Action to take 选项区中选择 Drop 单选按钮，如图 15-64 所示。在 Source address or network 标签后的下拉列表框中选择 Equals 选项，后面的对话框设为全部子网主机，即 192.168.10.*网段内的所有主机，接着在 Network protocol 标签后的下拉列表框中选择 Equals 命令，后面的协议类型选择 TCP，再将 Destination TCP or UDP port 选项，即目标地址的端口，设为 FTP 的端口 21，如图 15-65 所示。这样创建的规则的意思就是：所有 192.168.10.*网段内的主机发送到 21 号 TCP 端口的包全部丢弃，从而达到禁止子网客户机使用 FTP 的目的。

Chain and action details	
Part of chain	Incoming packets (INPUT)
Rule comment	deny FTP
Action to take	○ Do nothing ○ Accept ⊙ Drop ○ Reject ○ Userspace ○ Exit chain ○ Log packet ○ Run chain
Reject with ICMP type	⊙ 默认 ○ Type icmp-net-unreachable

图 15-64　为禁止 FTP 添加动作

Condition details		
Source address or network	Equals	192.168.10.0/24
Destination address or network	<Ignored>	
Incoming interface	<Ignored>	eth0
Outgoing interface	<Ignored>	eth0
Fragmentation	⊙ Ignored ○ Is fragmented ○ Is not fragmented	
Network protocol	Equals	TCP
Source TCP or UDP port	<Ignored>	⊙ Port(s) ○ Port range to
Destination TCP or UDP port	Equals	⊙ Port(s) 21 ○ Port range to

图 15-65　为禁止 FTP 添加条件

❹ 接着添加禁止 ICMP 协议的规则。还是在 FORWARD 链表格中，单击 Add Rule 按钮，弹出添加规则窗口，在 Action to take 选项区中选择 Drop 单选按钮，如图 15-66 所示。然后只要在 Network protocol 标签后的下拉列表框中选择 Equals 选项，后面的协议类型选择 ICMP，如图 15-67 所示，即可禁止 ICMP 的 IP 报文。

Chain and action details	
Part of chain	Incoming packets (INPUT)
Rule comment	deny ICMP
Action to take	○ Do nothing ○ Accept ⊙ Drop ○ Reject ○ Userspace ○ Exit chain ○ Log packet ○ Run chain
Reject with ICMP type	⊙ 默认 ○ Type icmp-net-unreachable

图 15-66　为禁止 ICMP 添加动作

Condition details			
Source address or network	<Ignored>		
Destination address or network	<Ignored>		
Incoming interface	Equals	eth0	
Outgoing interface	<Ignored>	eth0	
Fragmentation	⊙ Ignored ○ Is fragmented ○ Is not fragmented		
Network protocol	Equals	ICMP	

图 15-67　禁止 ICMP

5 最后讲述如何利用 NAT 表实现 IP 包的重定向功能。比如因为 Web 服务器的端口的改变，我们需要将由外部传入端口为 80 的 IP 包都重定向到 8080 端口，由第 3 章的知识可知，这条规则是添加在 NAT 表的 POSTROUTING 链上。在 Webmin 的防火墙设置的首页上，Show IPtables 后的下拉列表框中选择 Network address translation (nat)选项，然后单击 Show IPtables 按钮，出现 NAT 表三条链的表格，在 POSTROUTING 链中单击 Add Rule 按钮添加新规则，此时需在 Action to take 选项区中选中 Redirect 单选按钮，并将 Target ports for redirect 设为 8080，即重定向的目标端口，如图 15-68 所示。在规则的条件设置部分，将 Incoming interface 文本框设为 Equals、eth1，Destination TCP or UDP port 文本框设为 Equals、80，如图 15-69 所示，表示所有目标为 80 的从网卡 eth1 传入的 IP 包都送到目标地址是 8080 的端口，相当于我们在第 3 章中所添加的如下规则：

```
iptables-t nat -A PREROUTING -p tcp -i eth1 --dport 80 -j REDIRECT -to-ports
8080
```

Chain and action details	
Part of chain	Packets before routing (PREROUTING)
Rule comment	Redirect IP packets to Port 80
Action to take	○ Do nothing ○ Accept ○ Drop ⊙ Redirect ○ Destination NAT ○ Run chain
Target ports for redirect	○ 默认 ⊙ Port range 8080 to
IPs and ports for DNAT	⊙ 默认 ○ IP range to Port range to

图 15-68　添加重定向的动作

Condition details			
Source address or network	<Ignored>		
Destination address or network	<Ignored>		
Incoming interface	Equals	eth1	
Outgoing interface	<Ignored>	eth0	
Fragmentation	⊙ Ignored ○ Is fragmented ○ Is not fragmented		
Network protocol	Equals	TCP	
Source TCP or UDP port	<Ignored>	⊙ Port(s)	○ Port range to
Destination TCP or UDP port	Equals	⊙ Port(s) 80	○ Port range to
Source and destination port(s)	<Ignored>		

图 15-69　添加重定向的条件

📖**点评与拓展：** Webmin 全面提供了 Linux 防火墙的管理功能，完整列出了 iptables 的 3 张表中所有链的规则，并可以对其进行添加、删除、编辑等操作，大大简化了用户对防火墙的配置工作。

15.10　使用 Webmin 配置 MySQL 服务器

应用实例导航——利用 Webmin 架设 MySQL 服务器

※场景呈现

S 学校利用 Webmin 架设 MySQL 数据库服务器，通过 Webmin 提供的模块设置启动和关闭 MySQL 服务器，然后创建 person 数据库，并在其中创建 information 表，示例性地在 information 表中添加记录。

创建 guest 用户，使其在指定主机上远程登录 MySQL 数据库服务器。

※技术要领

(1) 理解关系数据库的构成。
(2) 利用 Webmin 进行 MySQL 的模块配置。
(3) 利用 Webmin 添加数据库、表和记录。
(4) 利用 Webmin 进行 MySQL 服务器的用户管理。

本节讲述如何在图形管理工具 Webmin 下面设置 MySQL 服务，本节中的例子以前面相关章节的部分实例为基础，具体参数设置的作用可以参见相关章节的内容。

❶ 在 Webmin 管理窗口，选择目录区中 Networking→MySQL Database Server 选项，打开 MySQL 数据库服务器管理窗口。图 15-70 显示了 MySQL 服务器尚未启动的页面，如果 mysqld 进程已经启动，则将直接进入 MySQL 数据库的管理页面。

❷ Webmin 默认配置是针对 Fedora 6 初始安装的 MySQL 数据库的，如果曾经重新安装过 MySQL 数据库，那么就需要在 Webmin 的【模块配置】窗口重新设定安装相关的参数。

这里以第 11 章中所讲述的安装在/usr/local/mysql 目录下的 MySQL 为例，单击图 15-70 中的【模块配置】按钮，打开 System configuration 窗口，如图 15-71 所示。将【mysql 命令的路径】文本框仍然设为/usr/bin/mysql，这里设置的是 MySQL 数据库登录命令 mysql 的存放路径，Webmin 根据此设置找到 MySQL 的入口程序，由于不管何种安装都将 MySQL 的入口程序复制在/usr/bin 目录下，因此此项无需改动。

需要注意到的是【启动 mysql 服务的命令】文本框的设置，由于 MySQL 数据库安装时没有将启动命令复制到/usr/bin 目录下，因此单击图 15-70 窗口中的【启动 MySQL 服务器】按钮时会因为系统找不到 mysqld 文件而无法启动。解决的一种办法是将 mysqld_safe 复制到/usr/bin 下，在【启动 mysql 服务的命令】文本框中写入 mysqld_safe-user=mysql。如果不将

mysqld_safe 复制到/usr/bin 目录下，则需要在【启动 mysql 服务的命令】文本框中写入/etc/local/mysql/bin/mysqld_safe --user=mysql &，其他项按图中所示进行设置。单击【保存】按钮，则将返回如图 15-70 所示的 MySQL 服务器管理窗口，单击【启动 MySQL 服务器】按钮就可以启动 MySQL 数据库服务器了。

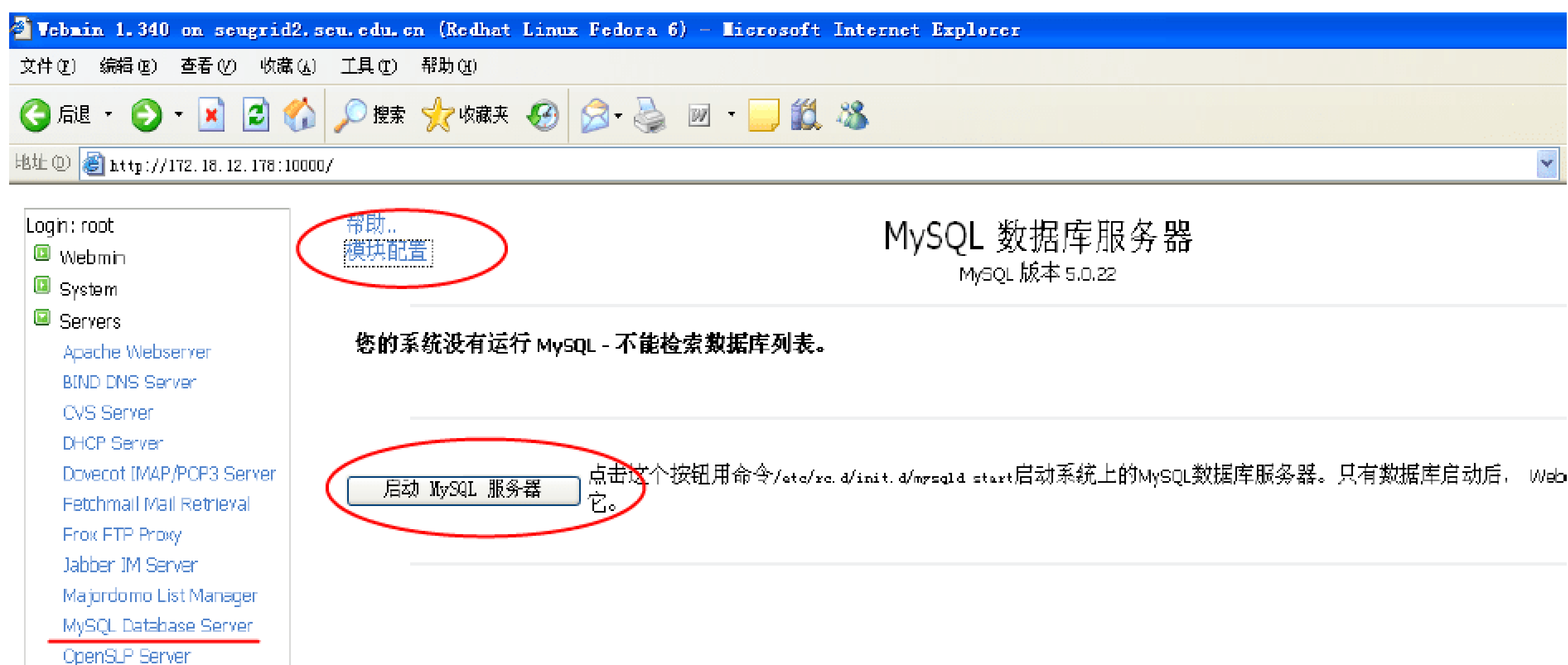


图 15-70　MySQL 数据库服务器设置窗口

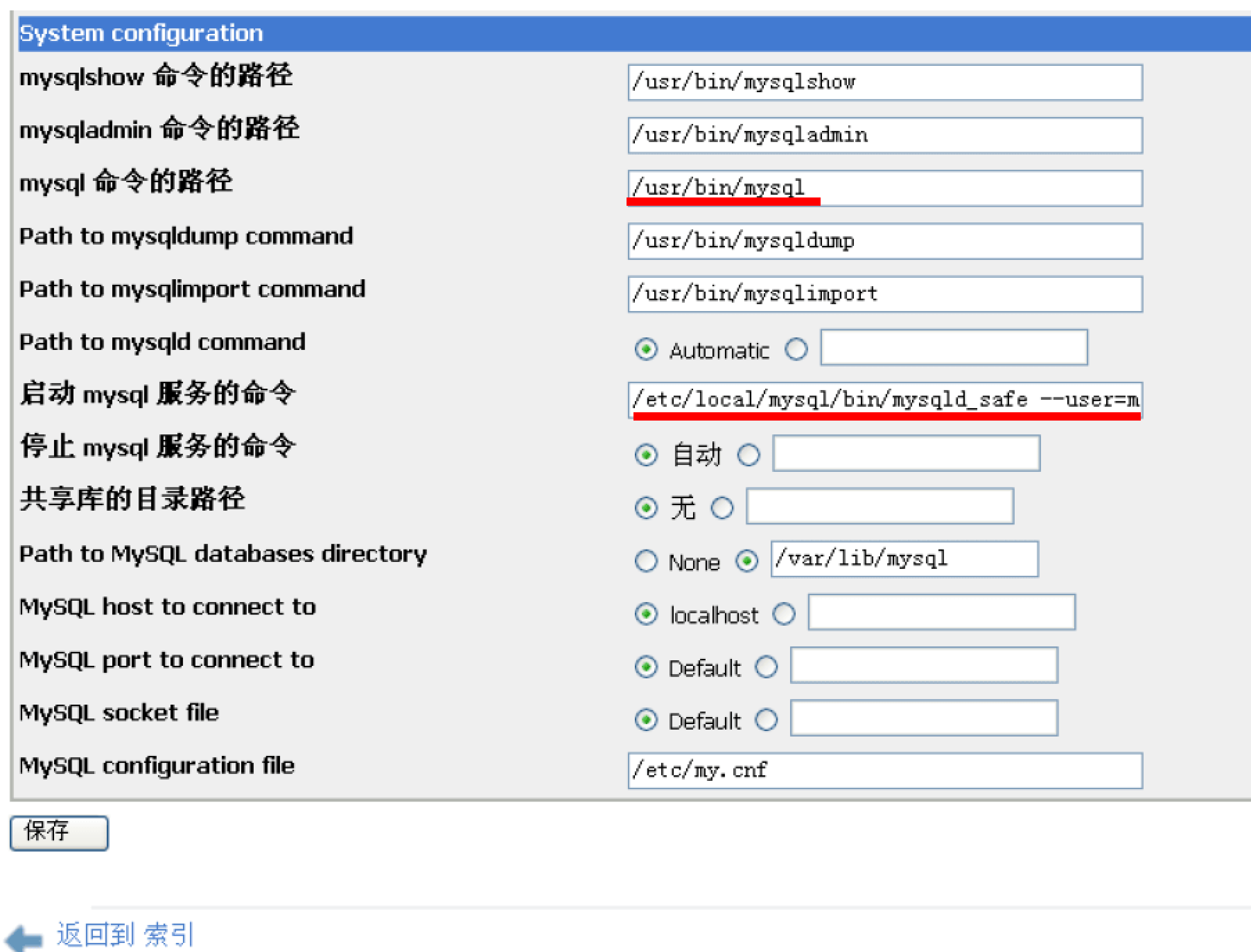


图 15-71　模块配置的设置

❸ 启动 MySQL 服务器之后，出现如图 15-72 所示窗口。窗口上方显示了 MySQL 现有的 4 个

数据库名称，如图 15-72 中标注部分所示，可以在此处对数据库表进行新建、删除和编辑操作。窗口下方提供了对 MySQL 服务器进行全局设定功能，即设置 MySQL 服务器相关的全局参数，主要是对各种权限的设置。

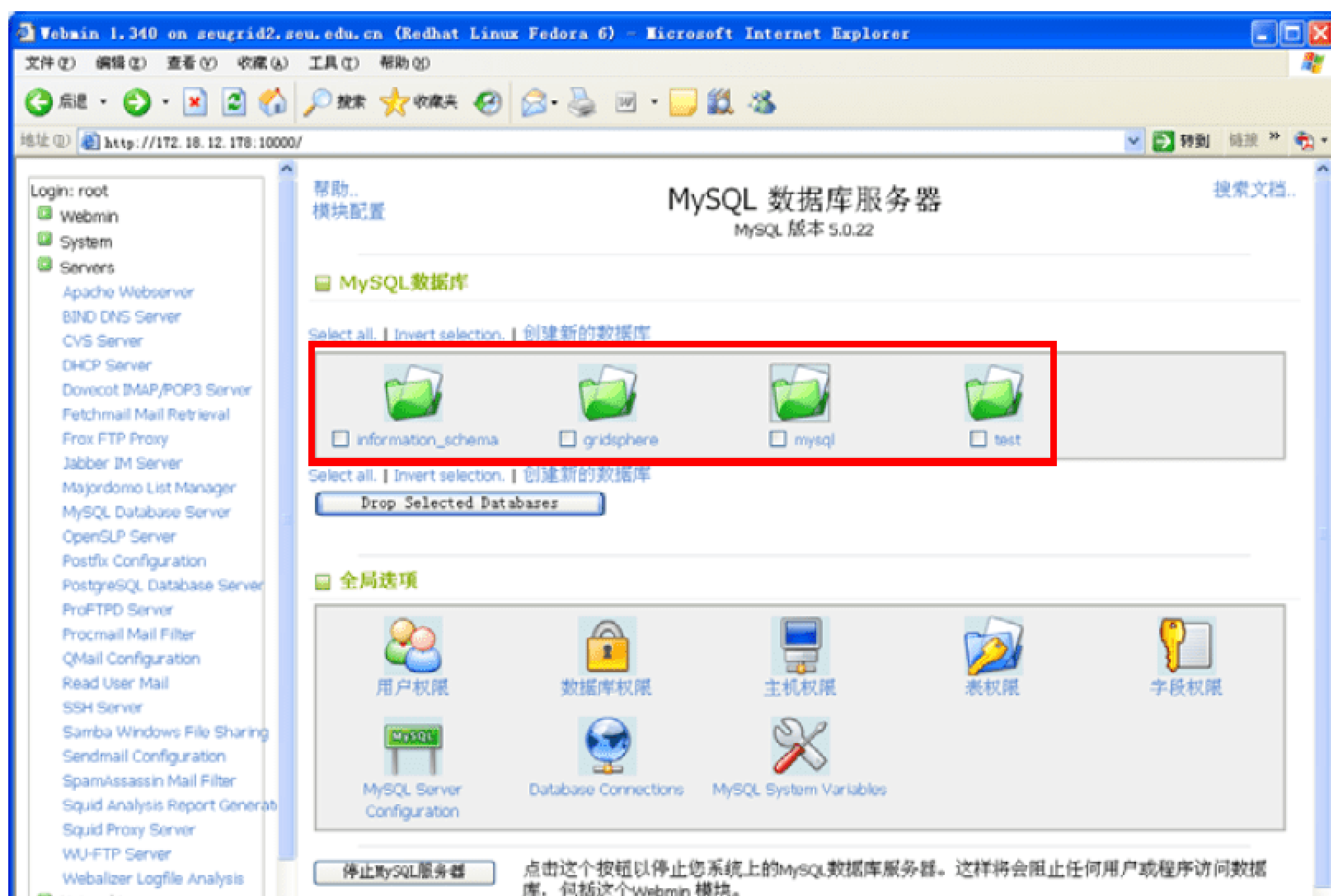


图 15-72　启动 MySQL 服务器后的管理窗口

❹ 创建 person 数据库只需单击【创建新的数据库】按钮，打开【新的数据库选项】窗口，如图 15-73 所示。在【数据库名】文本框中输入 person，如果需要为 person 数据库建立初始数据表，在【初始数据表】选项区中选中 Named 单选按钮，并为初始表设定名字和字段。本例只创建一个空的数据库，其他选项接受默认的设置，直接单击【新建】按钮确定。

Module 索引
帮助..

创建数据库

新的数据库选项

数据库名　person

Character set　<默认>

初始数据表　◉ 无　○ Named　with fields below

字段名	数据类型	类型宽度	主键?	自动增量?	允许空白	无符号的	默认值
			☐ 是	☐ 是	☑ 是	☐ 是	
			☐ 是	☐ 是	☑ 是	☐ 是	
			☐ 是	☐ 是	☑ 是	☐ 是	
			☐ 是	☐ 是	☑ 是	☐ 是	

新建

返回到 数据库列表

图 15-73　创建新的数据库 person

❺ person 数据库创建成功后，在图 15-72 的方框区域就会出现 person 数据库的图标，单击这

个图标进入【编辑数据库】窗口，如图 15-74 所示，该数据库没有任何表。如果要创建新表，在【新建新表】按钮后的【字段】文本框中输入表中字段的数目，然后单击【新建新表】打开【创建表】窗口，如图 15-75 所示。在【表名】文本框中输入 information，然后输入字段的名字、数据类型及宽度，我们选择 no 字段名为主键，需要注意的是，在使用 Webmin 创建新表时，主键是必须要设置的，否则表将创建不成功。如果所要创建的表没有合适的字段作为主键，可以创建一个与内容无关的序列号作为主键，并选中【自动增量】列中的复选框，这样该序列号会随着数据记录的增多而自动增加。但不能选择【允许空白】列中的复选框，即主键不能为空。information 表的其他字段的设置如图 15-75 所示，然后单击【新建】按钮创建新表。



图 15-74　数据库编辑窗口



图 15-75　创建新表 information

❻ information 表创建成功后，则在 person 数据库编辑窗口中出现 information 表的图标，如图 17-76 所示。单击该图标打开【编辑数据表】窗口，如图 15-77 所示，该页面列出了 information 表中各字段的详细信息，可以选择字段进行编辑或删除操作。

❼ 如果要为表添加记录或查看记录，则单击【查看数据】按钮打开【表数据】窗口，初始状态该页面显示表中无任何数据，单击【增加行】按钮将记录添加到页面，如图 15-78 所示。本例在 no 字段文本框中输入 0071，name 字段文本框中输入 BOB，sex 字段文本框中输入 m，birthday 字段文本框中输入 1986-09-11，phonenum 字段文本框中输入 025-8374313，address 字段文本框中输入 SEU，然后单击【保存】就添加了一条新的记录，在【表数据】编辑窗口显示了所创建的 BOB 记录，如图 15-79 所示。



图 15-76　创建新表 information 后的数据表列表



图 15-77　表 information 中各字段的详细信息



图 15-78　编辑表中数据

Module 索引

表数据

在数据库中person中的表information

Select all. | Invert selection.

	no	name	sex	birthday	phonenum	address
☐	0071	BOB	m	1986-09-11	025-8374313	SEU

Select all. | Invert selection.

编辑已选择的行 增加行 删除已选择的行

返回到 字段列表 | 返回到 数据表列表 | 返回到 数据库列表

图 15-79 新建的记录

❽ 接着我们讲述如何利用 Webmin 对用户、表和主机的权限进行设置。在【MySQL 数据库服务器】管理窗口中单击【全局选项】选项区中的【用户权限】图标，如图 15-80 所示。打开【用户权限】窗口，如图 15-81 所示，该页显示了用户的权限列表，【用户】项是 MySQL 数据库的用户，【主机】项指定该用户允许从哪台主机上登录，【加密密码】项存放了加密过后的用户密码，【权限】规定了该用户拥有的对数据库表的操作权限。



图 15-80 用户权限的设置

Module 索引
帮助..

用户权限

Select all. | Invert selection. | 创建新用户

	用户	主机	加密密码	权限
☐	匿名	seugrid2.seu.edu.cn		无
☐	匿名	localhost		无
☐	root	localhost		全部
☐	root	seugrid2.seu.edu.cn		全部
☐	seugrid	任何	*A05ACE78A574B7D63E18D4F155F3B09BC1470797	全部

Select all. | Invert selection. | 创建新用户

Delete Selected

下面的选项配置在Webmin中创建的 Unix用户与MySQL用户间的同步。

☐ 在新增一位Unix用户时也新增一位MySQL用户，包含权限...
☐ 在匹配的Unix用户改变后，也更新MySQL用户。
☐ 在匹配的Unix用户被删除后，也删除MySQL用户。

选择表数据
插入表数据
更新表数据
删除表数据
创建表

创建带主机名的新用户 ⦿ 全部主机 ○ 指定的主机

保存

返回到 数据库列表

图 15-81 【用户权限】窗口

❾ 如果需要创建新的 MySQL 数据库用户，比如创建一个 guest 用户，并只允许在

seugrid3.seu.edy.cn 这台主机上登录，则单击【创建新用户】按钮打开如图 15-82 所示窗口，在【用户名】选项区中选择【匿名用户】后的单选按钮，并输入 guest。【主机】选项区中选择【任何】后面的单选按钮，并输入 seugrid3.seu.edu.cn，如果将【主机】设为【任何】就表示允许该用户从任意主机上远程登录 MySQL 服务器。在【权限】下拉列表框中选择图中所示的 6 个权限。单击【保存】按钮，就创建一个设置了相应权限的新用户 guest，并显示在【用户权限】窗口中，如图 15-83 所示。

Module 索引
帮助..

MySQL用户详细信息

用户名	○ 匿名用户 ⦿ guest
密码	○ 无 ⦿ 设置为...
主机	○ 任何 ⦿ seugrid3.seu.edu.cn
权限	选择表数据 插入表数据 更新表数据 删除表数据 创建表 删除表 重载授权 关闭数据库 管理进程 文件操作

保存

图 15-82　创建成功新的 MySQL 用户 guest

用户	主机	加密密码	权限
匿名	seugrid2.seu.edu.cn		无
匿名	localhost		无
guest	seugrid3.seu.edu.cn		选择 | 插入 | 更新 | 删除 | 创建 | 删除
root	localhost		全部
root	seugrid2.seu.edu.cn		全部
seugrid	任何	*A05ACE78A574B7D63E18D4F155F3B098C1470797	全部

图 15-83　创建成功的 guest 用户

点评与拓展： 对于 MySQL 数据库的管理，一般不使用 shell 命令，通常可以使用 MySQL-Front 和 Webmin 两种软件所提供的图形化窗口。

15.11　使用 Webmin 管理系统软件

Webmin 还提供了 Linux 系统软件的安装、卸载等功能。一般情况下，系统软件的安装用 shell 命令完成，前面章节也讲过许多种软件安装的方法，但是系统软件的卸载在 shell 命令下往往难以很干净地删除该软件的所有相关文件，因此利用 Webmin 来卸载系统软件显得更加方便和实用。

❶ 在 Webmin 管理窗口中，单击目录区中 System→Software Packages 选项，弹出如图 15-84 所示的软件包管理窗口。主要分为两块：最上面的搜索软件包部分可以搜索系统已经安装好的软件包，并且提供显示系统软件包的树形结构。单击图 15-84 中的 Package Tree 按钮，弹出如图 15-85 所示的软件包树，它按功能将系统软件列出，方便用户的查找。中间一块是用于安装系统软件的模块，但是只支持 rpm 包的安装。rpm 包可以定位到 Linux 主机或是从 Windows 主机直接上传，甚至可以从 FTP 下载，这为用户提供了一定的便利。



图 15-84　Webmin 的软件包管理窗口

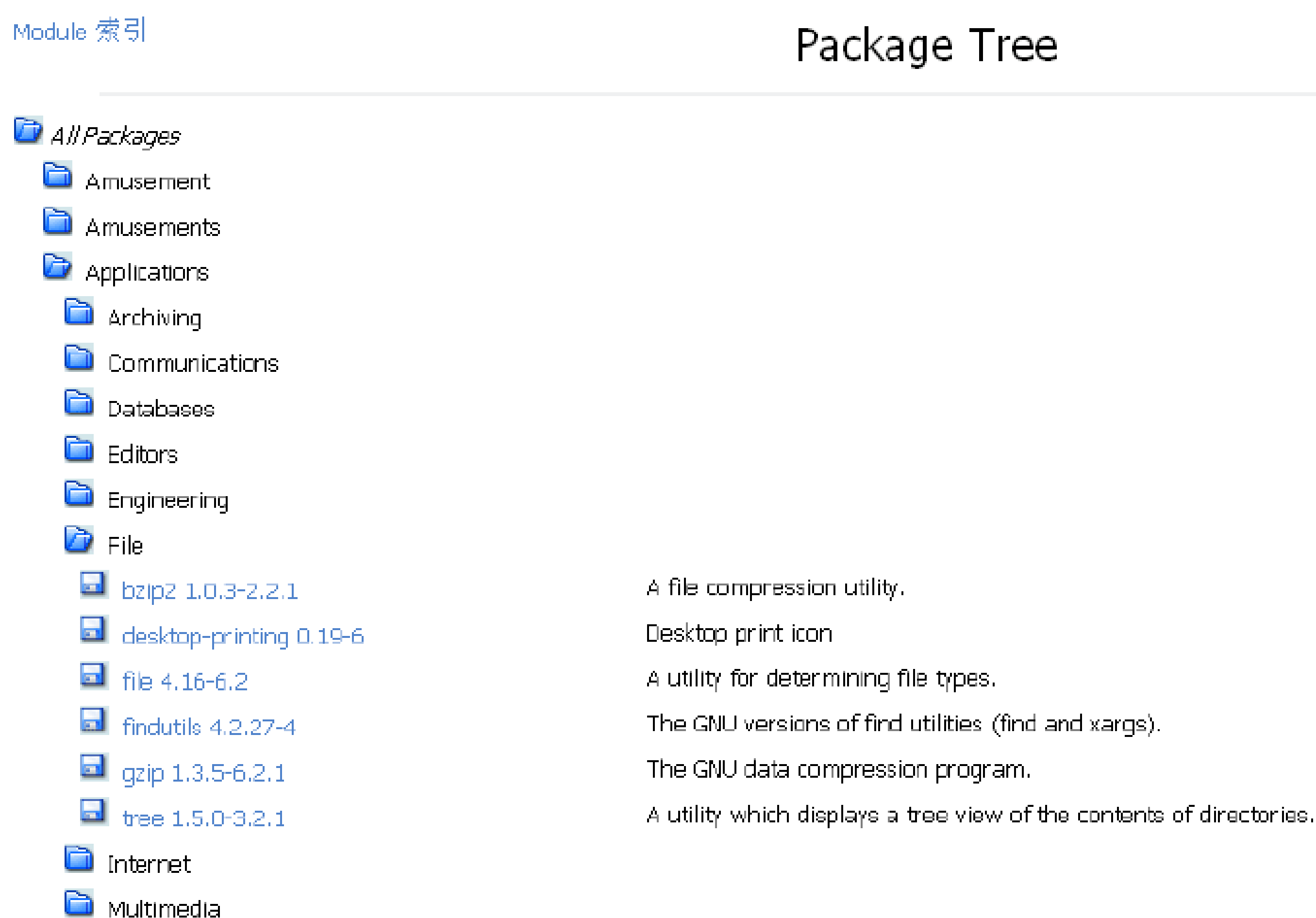


图 15-85　Linux 系统软件包树

❷ 如何使用 Webmin 卸载系统软件呢？比如我们需要卸载 sendmail 这个软件，则在软件包管理首页输入 sendmail，单击【搜索软件包】按钮，如图 15-84 标注部分所示。Webmin 会将所搜索到的关于 sendmail 的所有软件包列出来，如图 15-86 所示。图中一共有两个 sendmail 相关的包，选中这两个软件包，单击 Uninstall selected packages 按钮，弹出如图 15-87 所示的窗口。在 Ignore dependencies 选项区中选中【是】单选按钮，即忽略该软件与其他软件的依赖关系；如果选中【否】单选按钮，可能会由于其他软件使用 sendmail 软件包而导致无法删除。并在 Skip uninstall scripts 选项区中选中【是】单选按钮，如图 15-87 标注部分所示。然

后单击 Delete them all 按钮，即可删除所有 sendmail 相关的软件包。



图 15-86　搜索 sendmail 的结果

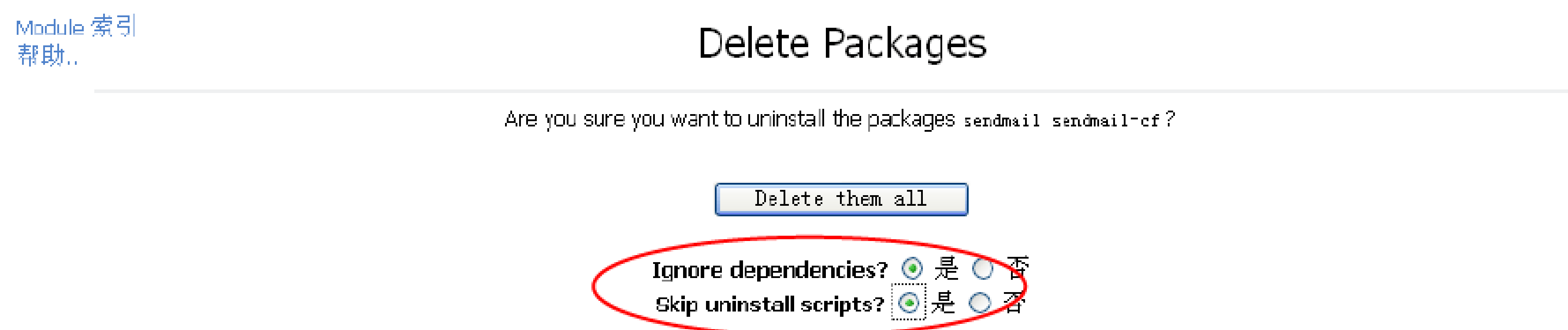


图 15-87　删除软件包

15.12　本 章 小 结

Webmin 是功能强大的基于 Web 的集系统管理和网络管理于一身的图形化 Linux 管理工具，网络管理员通过浏览器访问 Webmin 友好的用户窗口，从而轻松地管理本地或远程服务器。本章主要介绍了 Linux 图形化管理工具 Webmin 的安装和配置，以及利用 Webmin 管理图形化窗口分别对 DHCP、Samba、DNS、Web、NFS、SSH、MySQL、防火墙等服务的配置方法，以及利用 Webmin 管理系统软件。但是 Webmin 提供的仅仅是对各种服务的配置窗口，只能是方便网络管理员进行某些配置工作，而不可能替代使用 shell 命令行所进行的配置，因此建议读者在清晰理解各种服务器配置参数的意义的基础上，再辅以 Webmin 图形化工具来配置 Linux 的各种服务器。

第 16 章　Linux 服务器的性能监控

服务器性能监控就是为远程用户提供资源信息、资源状态等性能参数，并对资源进行监测、控制，有利于分析服务器性能瓶颈、发现服务器故障、及时恢复或调整系统。因此，随着目前公司或学术机构拥有的服务器数量日益增多，服务器性能监控技术成为了架设 Linux 时必须要考虑的重要问题之一。本章首先介绍服务器性能监控的意义，以及国际上流行的服务器性能监控软件 Ganglia 的应用现状；然后介绍 Ganglia 的安装、启动和测试；最后通过大型的案例，结合 Ganglia、MySQL、shell 脚本、Java、Web 服务器及 JSP 开发等前面章节的知识和实用技术，实现一个网络服务器性能监控系统，以 Web 页面的形式展示给终端用户。

通过本章的学习，读者应掌握以下内容：

- ✧ 理解服务器性能监控的概念和意义
- ✧ Ganglia 的安装和启动
- ✧ 利用 Ganglia 获取服务器的性能参数
- ✧ 理解网络服务器性能监控系统的设计和开发思路
- ✧ 掌握 shell 编程、Java/JSP 编程及 SQL 语言的综合使用

16.1　服务器性能监控概述

16.1.1　服务器性能监控的意义

服务器性能监控，就是在网络环境下为管理系统及终端用户提供性能参数信息、服务器状态等，所收集到的性能参数可以为管理系统制定决策、分配和调度资源等提供依据，并利于第一时间发现服务器故障，及时恢复调整服务器系统。长期对服务器性能进行监控所获得的数据，可以用来分析服务器性能的瓶颈、观察用户的行为，有助于优化服务器配置和部署。有鉴于此，国际上许多大学、著名 IT 公司、高性能计算中心等部门都越来越重视服务器的性能监控，普遍在部门的服务器上部署了监控软件，并提供给用户图形化的访问界面。因此，架设好的 Linux 服务器除了要考虑其系统安全性之外，如何监控其性能、监控其哪些性能参数、如何及时更新这些性能参数及以何种形式展示给用户都是必须要考虑的重要问题。服务器性能监控的研究逐渐成为激动人心的研究领域，设计和开发完善、稳定、实用的商用性能监控系统也必将成为颇具市场潜力的软件产品。

在利用 Windows NT/2K 所架设的服务器中，perfmon 提供完备的单机状态和性能监控；

在 Linux 系统中，可以用 uname、sysinfo、vmstat、netstat、ps、top 等 shell 命令查看特定的系统信息，但它们都仅仅为本地用户提供系统的性能信息，不支持以 XML 等 Web 数据库格式显示这些性能信息，而且也不支持远程用户的获取和访问。分布式的网络服务器监控系统应该具备如下特点。

- ✧ 动态性：服务器的性能参数可以分为静态信息和动态信息两类，静态信息是服务器不随时间变化的一些固有属性，如操作系统类型、CPU 主频、物理内存大小、磁盘空间总量、交换区总量等；动态信息是服务器随时间的变化、所运行的进程的不同而不断改变的一些参数，如空闲 CPU、空闲磁盘、CPU 在某时间段内的平均负载、交换区剩余空间等。因此，监控系统既要收集网络服务器的静态信息，又要收集其动态信息。
- ✧ 安全性：由于网络服务器分布在不同的地理位置，并且有些服务器应用于关键领域，因此至少不能破坏其原有的安全性。
- ✧ 最小性：资源监控系统要尽可能少地耗费服务器的资源。

基于对监控系统特点的分析，网络服务器监控的主要难点有以下几点。

- ✧ 被监控资源的多样性：监控对象包含硬件、软件及外部设备等多种类型的资源，而且资源之间的关系也比较复杂。
- ✧ 被监控资源数量巨大：互联的服务器数量急剧增多，如此众多的资源无法放到一个平面内统一管理。
- ✧ 被监控资源不具有内在逻辑结构：目前的局域网与互联网基本都是任意互联的，其中的资源缺少内在逻辑结构的描述与限定。
- ✧ 被监控资源的动态性：被监控资源无时无刻不在发生变化，监控系统必须跟踪这些变化。

因此在设计网络服务器性能监控系统时需要考虑到服务器监控独有的特点和难点，并使其能在广域范围内可扩展，能包容异构资源，并能在命名和安全方面与其他的成熟中间件集成。

16.1.2　Ganglia 简介

Ganglia 是一种可扩展的分布式监控系统，主要用于监控各种网络服务器、集群系统等高性能计算机系统。Ganglia 是由加利福尼亚伯克利分校开发的开源软件，最初由美国的 NPACI 和自然基金(NSF)资助，目标是为国家普适计算基础设施——网格所建立的动态监控软件。Ganglia 监控系统由两个守护进程(Daemon)，分别是客户端 Ganglia Monitoring Daemon (gmond)和服务端 Ganglia Meta Daemon (gmetad)，以及 Ganglia PHP Web Frontend(基于 Web 的动态访问方式)组成。gmond 模块运行于所有需要被监视的结点上，负责收集本结点的 CPU 负载、内存用量、磁盘空间等系统信息。

由于 Ganglia 的 gmond 进程可以方便地收集各结点的资源信息，国际上许多大学、著名 IT 公司、高性能计算中心所开发的监控系统大都基于 gmond 进程开发。比如，网站 http://monitor.millennium.berkeley.edu即为加利福尼亚伯克利分校的监控系统，图 16-1 显示了它以图形化的形式展示给终端用户服务器的性能参数，它利用 Ganglia 的 gmond 进程获

得 CPU 和存储方面的性能，并以图形化的 Web 界面清晰地展示了过去一小时内的 CPU 负载和内存负载情况。这种网络服务器的监控模式不仅可以方便地监控本部门的网络服务器的工作情况，而且方便了终端用户对本部门网络服务器的了解，相信会越来越流行。

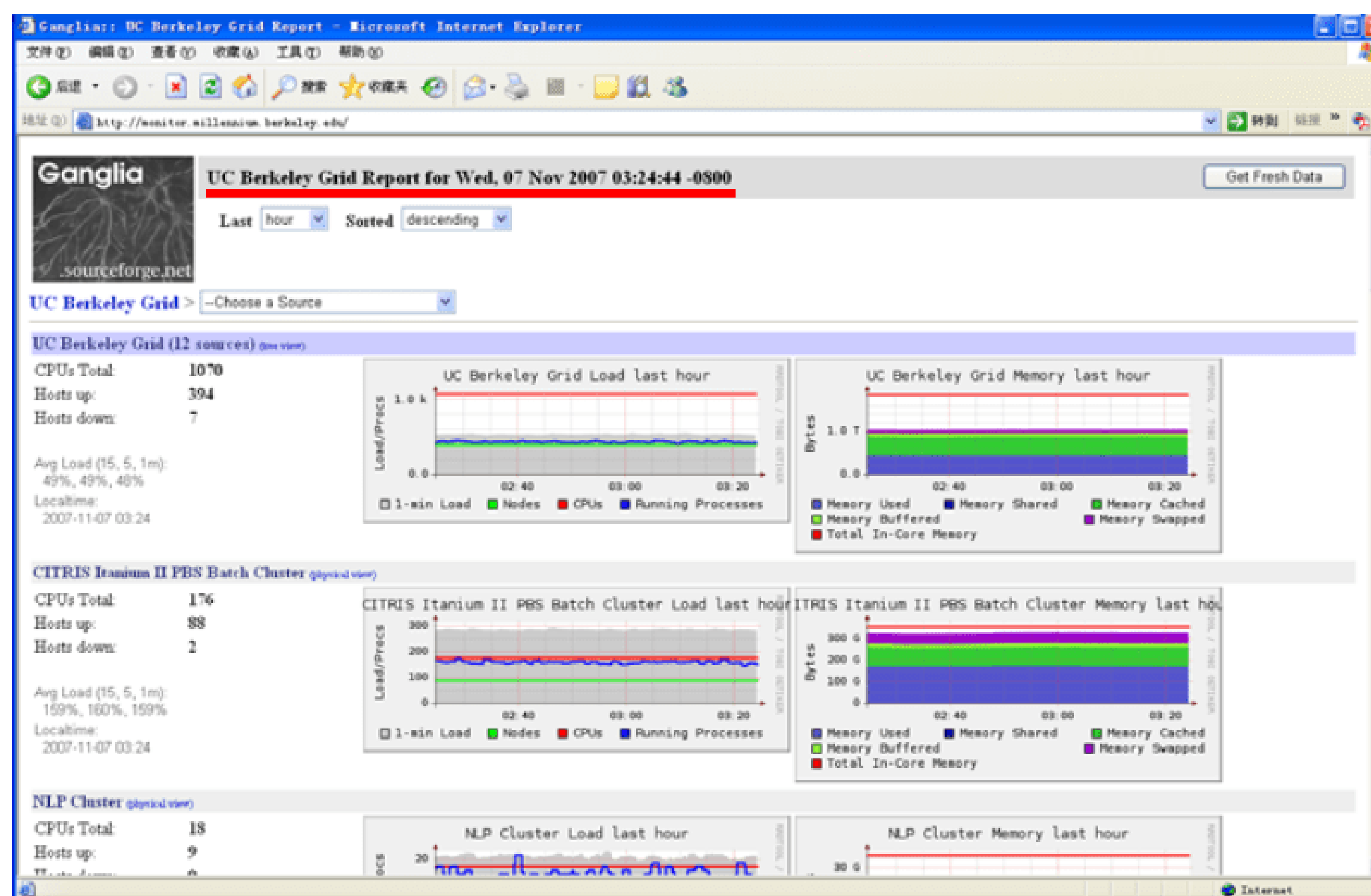


图 16-1　UC Berkeley 的监控系统

那么 gmond 进程具体可以获取哪些性能参数呢？它所收集的性能参数大概可以分为八组：CPU 信息、磁盘信息、过去一段时间内的平均负载、机器类型、内存信息、网络信息、操作系统信息和交换区信息，主要参数及对应描述如表 16-1 所示。

表 16-1　gmond 所收集的主要性能参数及描述

参数名称	描　述
cpu_idle	空闲 CPU 百分比
cpu_num	CPU 数目，实际上是 CPU 核的数目
cpu_speed	CPU 的主频，单位为 MHz
disk_free	总的磁盘空间
disk_total	空闲的磁盘空间
load_fifteen	15 分钟内的 CPU 平均负载
load_five	5 分钟内的 CPU 平均负载
load_one	1 分钟内的 CPU 平均负载
machine_type	机器类型，一般为 X_86_64
mem_buffers	缓冲区内存的大小
mem_cached	cache 的大小

续表

参数名称	描　述
mem_free	总的物理内存大小
mem_shared	总的虚拟内存大小
mem_total	总的内存大小，物理内存加上虚拟内存
mtu	网络最大的传输包的长度，单位是 B
os_name	操作系统名字
os_release	操作系统的发行版本
swap_free	空闲的交换区的空间
swap_total	总的交换区空间

16.2　Ganglia 的安装和部署

应用实例导航——Ganglia 的安装

※场景呈现

某高性能计算中心需要在多网卡的服务器上安装和部署 Ganglia 软件，并启动 gmond 进程，利用 Telnet 获取 gmond 所收集的服务器性能参数信息。

※技术要领

(1) Ganglia 的安装。
(2) gmond 的启动。
(3) 利用 Telnet 获取性能参数。

16.2.1　Ganglia 的安装

本节讲述 Ganglia 的安装过程。Ganglia 的安装包以源码形式提供，与前面章节所讲述的很多软件的安装过程一样，需要配置、编译、安装，具体步骤如下。

1. Ganglia 软件可以通过官方网站http://ganglia.sourceforge.net下载，如图 16-2 所示，单击图中标注处即可下载；或从本书附带的光盘获得，本节以 ganglia-3.0.4 版本为例示范安装，安装包名字为 ganglia-3.0.4.tar.gz，将安装包复制到/usr/local 目录，使用下面命令解压缩，得到/usr/local/ganglia-3.0.4 目录。

```
tar -zxvf ganglia-3.0.4.tar.gz
```



图 16-2　Ganglia 官方网站

❷ 进入/usr/local/ganglia-3.0.4 目录，安装包为源码结构，需要配置编译，使用下面命令配置 ganglia，如图 16-3 所示。

```
./configure
```

ganglia 包较大，配置需要一定时间，出现如图 16-4 所示的信息说明 ganglia 软件配置成功。

```
[root@seugrid2 local]# cd ganglia-3.0.4
[root@seugrid2 ganglia-3.0.4]# ls
acinclude.m4   config.h       COPYING            ganglia.spec       lib         mans        tests
aclocal.m4     config.h.in    ganglia.aix.spec   ganglia.spec.in    libmetrics  NEWS        web
AUTHORS        config.log     ganglia-config     gmetad             libtool     README
ChangeLog      config.status  ganglia-config.in  gmetric            Makefile    README.AIX
config         configure      ganglia.html       gmond              Makefile.am srclib
config.cache   configure.in   ganglia.pod        INSTALL            Makefile.in stamp-h1
[root@seugrid2 ganglia-3.0.4]# ./configure
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking build system type... i686-redhat-linux-gnu
checking host system type... i686-redhat-linux-gnu
```

图 16-3　配置 ganglia 软件

```
Welcome to..

Copyright (c) 2005 University of California, Berkeley

Version: 3.0.4 (Otto)
Library: Release 3.0.4 0:0:0

Type "make" to compile.
[root@seugrid2 ganglia-3.0.4]#
```

图 16-4　配置 ganglia 结束

❸ 配置结束后，可以使用下面命令编译，如图 16-5 所示。

```
make
```

```
[root@seugrid2 ganglia-3.0.4]# make
make  all-recursive
make[1]: Entering directory `/usr/local/ganglia-3.0.4'
Making all in lib
make[2]: Entering directory `/usr/local/ganglia-3.0.4/lib'
if /bin/sh ../libtool --tag=CC --mode=compile gcc -DHAVE_CONFIG_H -I. -
xpat/lib/ -I../libmetrics/ -I../srclib/apr/include/ -I../srclib/apr/inc
se/src -g -O2 -Wall -D_REENTRANT -MT ganglia.lo -MD -MP -MF ".deps/gang
.c; \
        then mv -f ".deps/ganglia.Tpo" ".deps/ganglia.Plo"; else rm -f
 gcc -DHAVE_CONFIG_H -I. -I. -I.. -I.. -I. -I../srclib/expat/lib/ -I../
de/ -I../srclib/apr/include/arch/unix/ -I../srclib/confuse/src -g -O2 -
-MD -MP -MF .deps/ganglia.Tpo -c ganglia.c  -fPIC -DPIC -o .libs/gangl
```

图 16-5　编译 ganglia

❹ ganglia 编译通过后，使用下面命令安装 ganglia，如图 16-6 所示。

```
make install
```

```
[root@seugrid2 ganglia-3.0.4]# make install
Making install in lib
make[1]: Entering directory `/usr/local/ganglia-3.0.4/lib'
make[2]: Entering directory `/usr/local/ganglia-3.0.4/lib'
test -z "/usr/lib" || mkdir -p -- "/usr/lib"
 /bin/sh ../libtool --mode=install /usr/bin/install -c  'libgar
/usr/bin/install -c .libs/libganglia-3.0.4.so.0.0.0 /usr/lib/l
(cd /usr/lib && rm -f libganglia-3.0.4.so.0 && ln -s libganglia
(cd /usr/lib && rm -f libganglia.so && ln -s libganglia-3.0.4.s
```

图 16-6　安装 Ganglia

16.2.2　Ganglia 的启动和测试

Ganglia 安装完毕后，就可以启动 Ganglia 的组件 gmond 了，并利用 gmond 进程获得服务器的各个性能参数。事实上，Ganglia 包括很多组件，包括客户端 Ganglia Monitoring Daemon (gmond)、服务端 Ganglia Meta Daemon (gmetad)和 Ganglia PHP Web Frontend(基于 Web 的动态访问方式)，本章仅使用到 gmond 组件，其他组件的功能读者可以参考 Ganglia 官方网站的相关介绍。

1. gmond 入口程序被默认安装在/usr/sbin 目录下，因此直接输入 gmond 就可以启动 gmond 进程。由于 gmond 进程是在 TCP 的 8649 端口侦听，故可以使用下面命令查看 gmond 的侦听状态，如图 16-7 所示。

```
netstat -apl | grep 8649
```

从图 16-7 可以看到，gmond 进程在所有网卡上都侦听，这也是 gmond 的优势所在。很多同等功能的服务器监控软件只能在一个网卡上侦听，导致某些网段内的信息难以监控。

```
[root@seugrid2 ~]# gmond
[root@seugrid2 ~]# netstat -apl | grep 8649
tcp        0      0 *:8649                    *:*                       LISTEN      30350/gmond
udp        0      0 seugrid2.se:filenet-rmi   239.2.11.71:8649          ESTABLISHED 30350/gmond
udp        0      0 239.2.11.71:8649          *:*                                   30350/gmond
[root@seugrid2 ~]#
```

图 16-7　启动 gmond 服务

2. 使用 Telnet 协议可以获得 Ganglia 所收集的性能参数，使用下面命令可以获得本机的性能参数，如图 16-8 所示。

```
telnet localhost 8649
```

图 16-8 中第二处标注表示，连接本地服务器获得性能参数。如果需要获取其他主机的性能参数，使用下面命令：

```
telnet IP 8649
```

比如，我们需要获得 172.18.12.178 的信息，输入“telnet 172.18.12.178 8649”，就尝试 telnet 连接远程主机，并获得其性能参数，如图 16-9 所示。但是，此时要确保 172.18.12.178 这台主机已经安装了 Telnet 服务器端软件，并成功启动 xinetd 服务，Telnet 的相关配置和启动的方法可以参考第 4 章“远程控制服务：Telnet、SSH 和 VNC”的论述。

3. Ganglia 所收集的所有性能参数如图 16-10 所示，它以 XML 格式显示，以标签<GANGLIA_XML_VERSION="3.0.4" SOURCE="gmond">开头，</GANGLIA_XML>结尾。图 16-11 显示了部分性能参数的详细信息，每一行每一对的“<METRIC>”内即为一个性能参数信息，NAME 是该性能参数的名字，比如 cpu_speed 即为 CPU 主频，load_five 即为 CPU 5 分钟内的负载，VAL 为该性能参数的数值，TYPE 为该数值的类型，UNITS 为该性能参数的单位，等等。从图 16-11 可以看到，CPU 主频数值为 2992，类型是字符型，单位是

MHz，表示主频为 3GHz。

```
[root@seugrid2 ~]# telnet localhost 8649
Trying 127.0.0.1...
Connected to localhost.localdomain (127.0.0.1).
Escape character is '^]'.
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<!DOCTYPE GANGLIA_XML [
   <!ELEMENT GANGLIA_XML (GRID|CLUSTER|HOST)*>
      <!ATTLIST GANGLIA_XML VERSION CDATA #REQUIRED>
      <!ATTLIST GANGLIA_XML SOURCE CDATA #REQUIRED>
   <!ELEMENT GRID (CLUSTER | GRID | HOSTS | METRICS)*>
      <!ATTLIST GRID NAME CDATA #REQUIRED>
      <!ATTLIST GRID AUTHORITY CDATA #REQUIRED>
      <!ATTLIST GRID LOCALTIME CDATA #IMPLIED>
   <!ELEMENT CLUSTER (HOST | HOSTS | METRICS)*>
      <!ATTLIST CLUSTER NAME CDATA #REQUIRED>
      <!ATTLIST CLUSTER OWNER CDATA #IMPLIED>
      <!ATTLIST CLUSTER LATLONG CDATA #IMPLIED>
      <!ATTLIST CLUSTER URL CDATA #IMPLIED>
      <!ATTLIST CLUSTER LOCALTIME CDATA #REQUIRED>
   <!ELEMENT HOST (METRIC)*>
      <!ATTLIST HOST NAME CDATA #REQUIRED>
      <!ATTLIST HOST IP CDATA #REQUIRED>
      <!ATTLIST HOST LOCATION CDATA #IMPLIED>
      <!ATTLIST HOST REPORTED CDATA #REQUIRED>
      <!ATTLIST HOST TN CDATA #IMPLIED>
      <!ATTLIST HOST TMAX CDATA #IMPLIED>
      <!ATTLIST HOST DMAX CDATA #IMPLIED>
      <!ATTLIST HOST GMOND_STARTED CDATA #IMPLIED>
```

图 16-8　获取本机服务器性能参数

```
[root@sec local]# telnet 172.18.12.178 8649
Trying 172.18.12.178...
Connected to seugrid2.seu.edu.cn (172.18.12.178).
Escape character is '^]'.
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<!DOCTYPE GANGLIA_XML [
   <!ELEMENT GANGLIA_XML (GRID|CLUSTER|HOST)*>
      <!ATTLIST GANGLIA_XML VERSION CDATA #REQUIRED>
      <!ATTLIST GANGLIA_XML SOURCE CDATA #REQUIRED>
   <!ELEMENT GRID (CLUSTER | GRID | HOSTS | METRICS)*>
      <!ATTLIST GRID NAME CDATA #REQUIRED>
      <!ATTLIST GRID AUTHORITY CDATA #REQUIRED>
      <!ATTLIST GRID LOCALTIME CDATA #IMPLIED>
   <!ELEMENT CLUSTER (HOST | HOSTS | METRICS)*>
      <!ATTLIST CLUSTER NAME CDATA #REQUIRED>
      <!ATTLIST CLUSTER OWNER CDATA #IMPLIED>
      <!ATTLIST CLUSTER LATLONG CDATA #IMPLIED>
      <!ATTLIST CLUSTER URL CDATA #IMPLIED>
      <!ATTLIST CLUSTER LOCALTIME CDATA #REQUIRED>
   <!ELEMENT HOST (METRIC)*>
```

图 16-9　获取其他主机的服务器性能参数

```
<GANGLIA_XML VERSION="3.0.4" SOURCE="gmond">
<CLUSTER NAME="unspecified" LOCALTIME="1194324427" OWNER="unspecified" LATLONG="unspecified" URL="unspecified">
<HOST NAME="seugrid2.seu.edu.cn" IP="172.18.12.178" REPORTED="1194324421" TN="5" TMAX="20" DMAX="0" LOCATION="unspecified" GM
OND_STARTED="1194324101">
<METRIC NAME="disk_total" VAL="25.618" TYPE="double" UNITS="GB" TN="320" TMAX="1200" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="cpu_speed" VAL="2992" TYPE="uint32" UNITS="MHz" TN="320" TMAX="1200" DMAX="0" SLOPE="zero" SOURCE="gmond"/>
<METRIC NAME="part_max_used" VAL="46.5" TYPE="float" UNITS="" TN="140" TMAX="180" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="swap_total" VAL="1048568" TYPE="uint32" UNITS="KB" TN="320" TMAX="1200" DMAX="0" SLOPE="zero" SOURCE="gmond"/>
<METRIC NAME="os_name" VAL="Linux" TYPE="string" UNITS="" TN="320" TMAX="1200" DMAX="0" SLOPE="zero" SOURCE="gmond"/>
<METRIC NAME="cpu_user" VAL="0.0" TYPE="float" UNITS="%" TN="50" TMAX="90" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="cpu_system" VAL="0.0" TYPE="float" UNITS="%" TN="50" TMAX="90" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="cpu_aidle" VAL="99.9" TYPE="float" UNITS="%" TN="50" TMAX="3800" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="load_five" VAL="0.00" TYPE="float" UNITS="" TN="50" TMAX="325" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="proc_run" VAL="0" TYPE="uint32" UNITS="" TN="0" TMAX="950" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="mem_free" VAL="13724" TYPE="uint32" UNITS="KB" TN="140" TMAX="180" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="mem_buffers" VAL="32908" TYPE="uint32" UNITS="KB" TN="140" TMAX="180" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="swap_free" VAL="1048436" TYPE="uint32" UNITS="KB" TN="140" TMAX="180" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="bytes_in" VAL="263.92" TYPE="float" UNITS="bytes/sec" TN="20" TMAX="300" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="pkts_out" VAL="1.33" TYPE="float" UNITS="packets/sec" TN="20" TMAX="300" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="cpu_num" VAL="2" TYPE="uint16" UNITS="CPUs" TN="320" TMAX="1200" DMAX="0" SLOPE="zero" SOURCE="gmond"/>
<METRIC NAME="disk_free" VAL="13.748" TYPE="double" UNITS="GB" TN="140" TMAX="180" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="mem_total" VAL="472516" TYPE="uint32" UNITS="KB" TN="320" TMAX="1200" DMAX="0" SLOPE="zero" SOURCE="gmond"/>
<METRIC NAME="cpu_wio" VAL="0.0" TYPE="float" UNITS="%" TN="50" TMAX="90" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="boottime" VAL="1193979757" TYPE="uint32" UNITS="s" TN="320" TMAX="1200" DMAX="0" SLOPE="zero" SOURCE="gmond"/>
<METRIC NAME="machine_type" VAL="x86" TYPE="string" UNITS="" TN="320" TMAX="1200" DMAX="0" SLOPE="zero" SOURCE="gmond"/>
<METRIC NAME="os_release" VAL="2.6.18-1.2798.fc6xen" TYPE="string" UNITS="" TN="320" TMAX="1200" DMAX="0" SLOPE="zero" SOURCE
="gmond"/>
<METRIC NAME="cpu_nice" VAL="0.0" TYPE="float" UNITS="%" TN="50" TMAX="90" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="cpu_idle" VAL="100.0" TYPE="float" UNITS="%" TN="50" TMAX="90" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="load_one" VAL="0.00" TYPE="float" UNITS="" TN="50" TMAX="70" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="load_fifteen" VAL="0.00" TYPE="float" UNITS="" TN="50" TMAX="950" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="proc_total" VAL="159" TYPE="uint32" UNITS="" TN="0" TMAX="950" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="mem_shared" VAL="0" TYPE="uint32" UNITS="KB" TN="140" TMAX="180" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="mem_cached" VAL="209288" TYPE="uint32" UNITS="KB" TN="140" TMAX="180" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="gexec" VAL="OFF" TYPE="string" UNITS="" TN="20" TMAX="300" DMAX="0" SLOPE="zero" SOURCE="gmond"/>
<METRIC NAME="bytes_out" VAL="102.63" TYPE="float" UNITS="bytes/sec" TN="20" TMAX="300" DMAX="0" SLOPE="both" SOURCE="gmond"/
>
<METRIC NAME="pkts_in" VAL="3.28" TYPE="float" UNITS="packets/sec" TN="20" TMAX="300" DMAX="0" SLOPE="both" SOURCE="gmond"/>
</HOST>
</CLUSTER>
</GANGLIA_XML>
```


图 16-10　Ganglia 所收集的性能参数


```
<METRIC NAME="disk_total" VAL="25.618" TYPE="double" UNITS="GB" TN="320" TMAX="1200" D
<METRIC NAME="cpu_speed" VAL="2992" TYPE="uint32" UNITS="MHz" TN="320" TMAX="1200" DMA
<METRIC NAME="part_max_used" VAL="46.5" TYPE="float" UNITS="" TN="140" TMAX="180" DMAX
<METRIC NAME="swap_total" VAL="1048568" TYPE="uint32" UNITS="KB" TN="320" TMAX="1200"
<METRIC NAME="os_name" VAL="Linux" TYPE="string" UNITS="" TN="320" TMAX="1200" DMAX="0
<METRIC NAME="cpu_user" VAL="0.0" TYPE="float" UNITS="%" TN="50" TMAX="90" DMAX="0" SL
<METRIC NAME="cpu_system" VAL="0.0" TYPE="float" UNITS="%" TN="50" TMAX="90" DMAX="0"
<METRIC NAME="cpu_aidle" VAL="99.9" TYPE="float" UNITS="%" TN="50" TMAX="3800" DMAX="0
<METRIC NAME="load_five" VAL="0.00" TYPE="float" UNITS="" TN="50" TMAX="325" DMAX="0"
```


图 16-11　具体的性能参数属性

📖**点评与拓展**：gmond 所收集的许多组性能参数中，有一些性能参数是静态不变的，如 os_name 操作系统名称，men_toal 物理内存总量等参数；但是有些参数是随着 gmond 收集时间的不同而改变的，如 disk_free 空闲磁盘空间，load_five 5 分钟内 CPU 负载等参数。

16.3　网络服务器的性能监控

应用实例导航——网络服务器监控系统的实现

※场景呈现

某高性能计算中心为了能提供给终端用户全面的服务器信息，并随时监控服务器的运

行状态和性能参数，需要收集部署在局域网内所有服务器的性能参数，将这些性能参数存放在中心数据库，并动态地实时更新。

中心数据库利用 MySQL 软件架设，由两张数据库表来存放资源信息，第一张表为 resourcetype，存放性能参数名称及其单位，这张表静态存在，无需更新；第二张表为 resources 表，存放各服务器的所有性能参数及参数的数值，这张表定时动态更新。服务器部署和网络拓扑如图 16-12 所示。假设该计算中心有两个局域网，网段分别为 192.168.10.*和 172.18.12.*，我们在两个局域网段内的服务器上都部署 Ganglia 软件，搜集性能参数，然后分别通过网关发送到 MySQL 数据库服务器，而 Web 服务器与 MySQL 数据库同步，以 jsp 或 html 网页的形式显示所有服务器的性能参数，提供给用户以 HTTP 协议的方式访问。

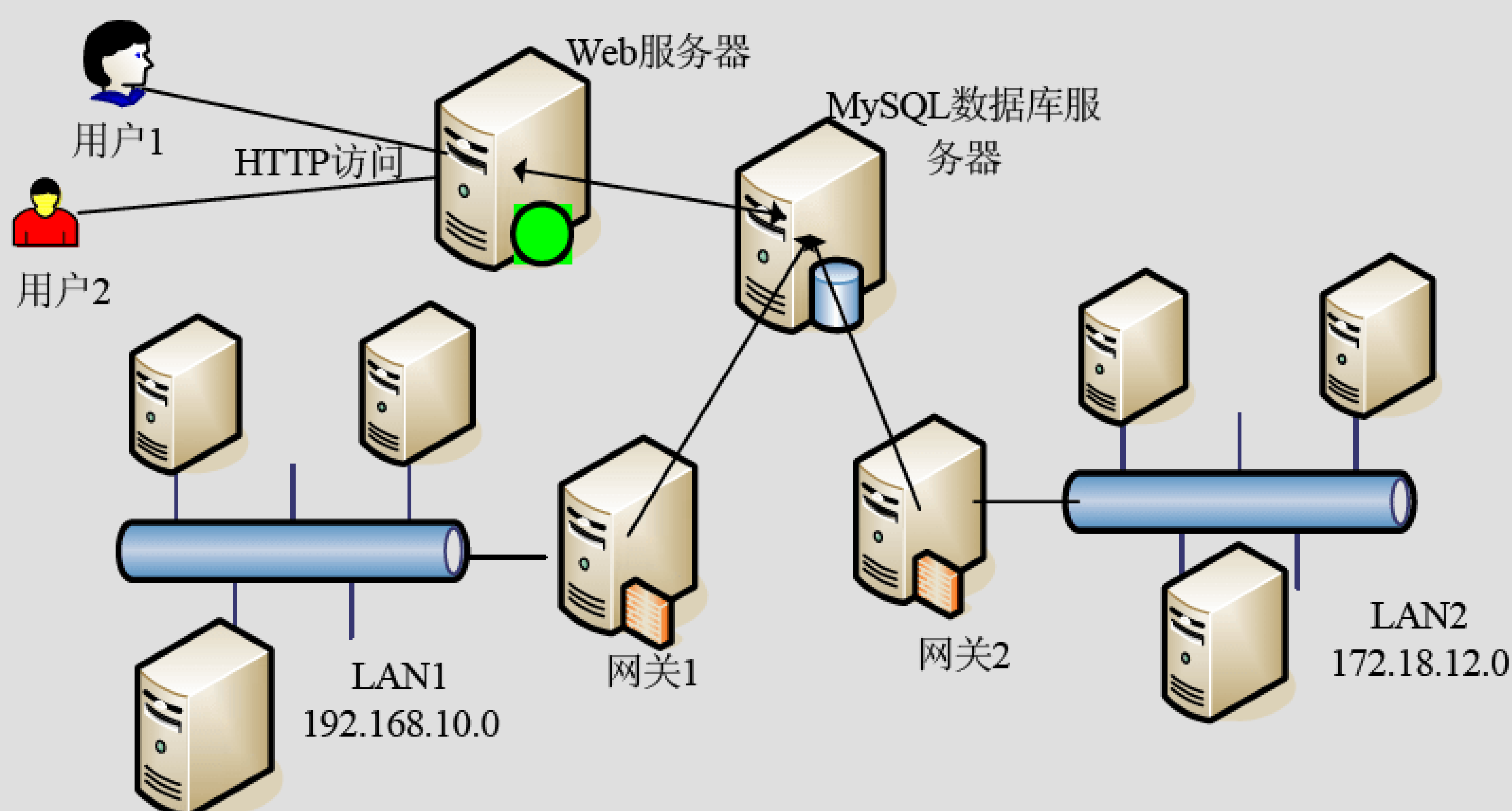


图 16-12　服务器部署示意图

※技术要领

(1) 利用 Ganglia 收集服务器性能参数。
(2) 利用 Java 程序解析 XML 格式文件。
(3) 利用 Java 程序建立、更新数据库。
(4) 利用 JSP 技术显示 MySQL 数据库信息。
(5) 利用 crontab 定时更新性能参数。

本节通过场景呈现中的实际案例来讲述如何利用 Ganglia 软件采集服务器的信息，并结合前面章节中所讲述的 MySQL 数据库和 Web 服务器的相关知识，将 Ganglia 所收集的性能参数存放到数据库表中，并提供给用户以 HTTP 协议的方式访问。本节将网络服务器监控系统的实现分为四个子问题：首先将 Ganglia 所收集的性能参数信息制成一个 XML 文件；然后利用 Java 程序解析该 XML 文件；接着将解析出的性能参数信息逐行写入 MySQL 数据库中；最后实现数据库中的信息定时更新，并通过 Web 页面图形化地展示给用户。

16.3.1 利用 Ganglia 生成 XML 文件

由于 Ganglia 软件本身就以 XML 的形式在 shell 上显示所收集到的性能参数信息，因此我们将 shell 信息重定向到文本文件就可以生成性能参数的 XML 文件了。

❶ 将性能参数的 XML 文件取名为 gmond_msg_1.txt，使用下面命令实现 shell 信息的重定向，如图 16-13 所示。

```
telnet localhost 8649 > gmond_msg_1.txt
```

shell 命令中的“>”符号的意思是将 shell 的输出结果写入到该符号后面的文件中，类似于 C++中的输出流“<<”符号，但是最后一行“Connection closed by foreign host.”将不写入 gmond_msg_1.txt 文件，仍然回显在 shell 上。

```
[root@seugrid2 etc]# telnet localhost 8649 > gmond_msg_1.txt
Connection closed by foreign host.
```

图 16-13　生成 gmond_msg_1.txt 文件

❷ 为了检验性能参数的 XML 文件是否成功写入 gmond_msg_1.txt 文件，可以使用下面命令查看 gmond_msg_1.txt 文件的最后 20 行。

```
tail -n 20 gmond_msg_1.txt
```

tail 命令经常用来查看长文件的最后几行的内容，如查看日志文件的最新信息也用该命令，-n 参数后带的数字表示查看最后多少行的内容。结果如图 16-14 所示，gmond_msg_1.txt 文件中成功记录了 gmond 所收集到的参数性能信息。

```
[root@seugrid2 etc]# tail -n 20 gmond_msg_1.txt
<METRIC NAME="cpu_num" VAL="2" TYPE="uint16" UNITS="CPUs" TN="60" TMAX="1200" DMAX="0" SLOPE="zero" SOURCE="gm
<METRIC NAME="disk_free" VAL="13.748" TYPE="double" UNITS="GB" TN="0" TMAX="180" DMAX="0" SLOPE="both" SOURCE=
<METRIC NAME="mem_total" VAL="472516" TYPE="uint32" UNITS="KB" TN="60" TMAX="1200" DMAX="0" SLOPE="zero" SOURC
<METRIC NAME="cpu_wio" VAL="0.0" TYPE="float" UNITS="%" TN="0" TMAX="90" DMAX="0" SLOPE="both" SOURCE="gmond"/
<METRIC NAME="boottime" VAL="1193979757" TYPE="uint32" UNITS="s" TN="60" TMAX="1200" DMAX="0" SLOPE="zero" SOU
<METRIC NAME="machine_type" VAL="x86" TYPE="string" UNITS="" TN="60" TMAX="1200" DMAX="0" SLOPE="zero" SOURCE=
<METRIC NAME="os_release" VAL="2.6.18-1.2798.fc6xen" TYPE="string" UNITS="" TN="60" TMAX="1200" DMAX="0" SLOPE
"gmond"/>
<METRIC NAME="cpu_nice" VAL="0.0" TYPE="float" UNITS="%" TN="0" TMAX="90" DMAX="0" SLOPE="both" SOURCE="gmond"
<METRIC NAME="cpu_idle" VAL="100.0" TYPE="float" UNITS="%" TN="0" TMAX="90" DMAX="0" SLOPE="both" SOURCE="gmon
<METRIC NAME="load_one" VAL="0.00" TYPE="float" UNITS="" TN="0" TMAX="70" DMAX="0" SLOPE="both" SOURCE="gmond"
<METRIC NAME="load_fifteen" VAL="0.00" TYPE="float" UNITS="" TN="0" TMAX="950" DMAX="0" SLOPE="both" SOURCE="g
<METRIC NAME="proc_total" VAL="159" TYPE="uint32" UNITS="" TN="60" TMAX="950" DMAX="0" SLOPE="both" SOURCE="gm
<METRIC NAME="mem_shared" VAL="0" TYPE="uint32" UNITS="KB" TN="100" TMAX="180" DMAX="0" SLOPE="both" SOURCE="g
<METRIC NAME="mem_cached" VAL="210132" TYPE="uint32" UNITS="KB" TN="100" TMAX="180" DMAX="0" SLOPE="both" SOUR
<METRIC NAME="gexec" VAL="OFF" TYPE="string" UNITS="" TN="60" TMAX="300" DMAX="0" SLOPE="zero" SOURCE="gmond"/
<METRIC NAME="bytes_out" VAL="136.98" TYPE="float" UNITS="bytes/sec" TN="60" TMAX="300" DMAX="0" SLOPE="both"
>
<METRIC NAME="pkts_in" VAL="4.28" TYPE="float" UNITS="packets/sec" TN="60" TMAX="300" DMAX="0" SLOPE="both" SC
</HOST>
</CLUSTER>
</GANGLIA_XML>
[root@seugrid2 etc]#
```

图 16-14　查看 gmond_msg_1.txt 文件

16.3.2 解析 XML 文件

gmond_msg_1.txt 文件以 XML 格式存放各性能参数，需要利用一个程序把性能参数的名字、数值、类型等有用的信息逐行地提取出来，由于每个性能参数都以“<METRIC”开

头，很有规律，因此利用 C 语言或 Java 语言的 I/O 类对文件流控制逐行读取 gmond_msg_1.txt 文件内容，识别“<METRIC”、VAL 标记等标记，从而提取出每个性能参数的名称、数值和类型之间的对应关系。

本书利用 Java 语言来实现，设计了一个类 ReadGmondXML.Java 对此 XML 文件的解析，提取每个性能参数的四个属性——名字、数值、类型和单位，存放在数据结构 ResourcePara 内，ResourcePara 的定义如图 16-15 所示。



图 16-15 保存解析后的性能参数的数据结构

如，对于资源 cpu_nice，它的 ResourcePara 对象的各成员的值为 NAME= "cpu_nice" ，VAL="0.0" ，TYPE= "float" ，UNITS= "%"。每种资源信息参数对应一个 ResourcePara 对象，总计 20 个对象。

有关结点资源信息采集的操作，即解析 XML 文件的操作，全部封装在 ReadGmondXML 类中，该类的属性和方法如图 16-16 所示。

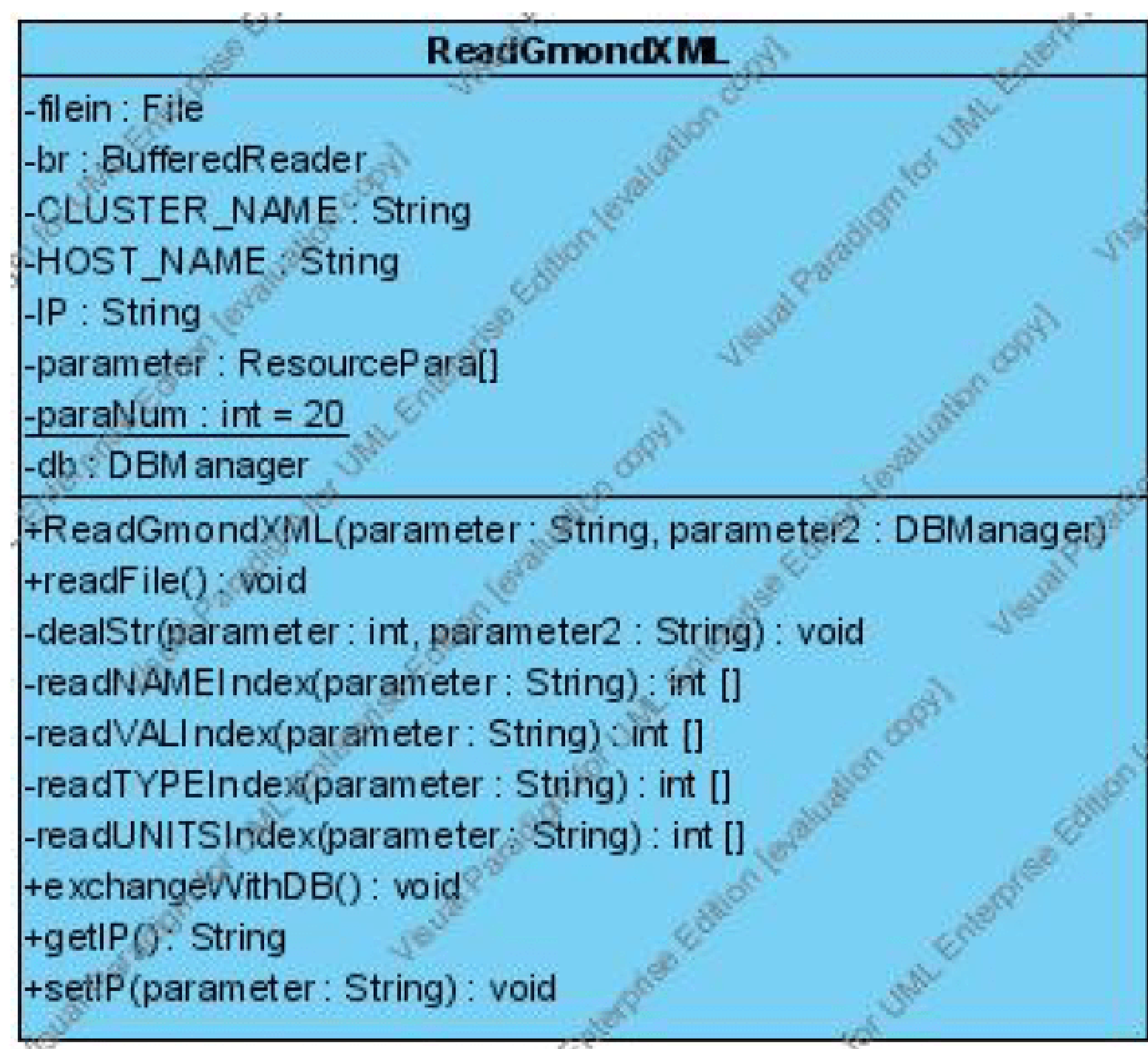


图 16-16 类 ReadGmondXML 成员

解析 XML 文件的步骤为：

(1) 在 ReadFile 函数中每次读取一行作为一个字符串。

(2) 在 ReadFile 函数中调用 dealStr 函数处理该字符串，当遇到 METRIC NAME、VAL、UNITS 子字符串时调用相应的 readNameIndex、readVALIndex、readUNITSIndex 函数，把值记录到 ResourcePara 对象相应的值中。ReadGmondXML 类请见本书附带光盘中的 ReadGmondXML.Java 文件，这里列出其中的关键代码加以解释。

```
/**
 * @author HuangHe
 * 从 gmond 产生的 XML 文件中提取服务器性能参数的四个属性，并将服务器性能参数存放到
ResourcePara 数据结构中
 */
public class ReadGmondXML {
//类的属性定义
private File filein;
private BufferedReader br;
private String CLUSTER_NAME=null;
private String HOST_NAME=null;
private String IP=null;
private ResourcePara parameter[];
private static final int paraNum=20;
private DBManager db;
//类的方法定义
public ReadGmondXML(String filename, DBManager db) {
    System.out.println("\n\n\n ----------ReadGmondXML---------- \n\n\n");
    this.db=db;
    parameter=new ResourcePara[paraNum];
    for(int i=0;i<paraNum;i++){
        parameter[i]=new ResourcePara("", "","","");
    }
    try {
        filein=new File(filename);
        br=new BufferedReader(new FileReader(filein));
    } catch (FileNotFoundException e) {
        System.err.println("cannot find the file\t 输入文件错误，请重新输入文
件名");
        e.printStackTrace();
    }
    readFile();
    exchangeWithDB();
```

```
        //this code must be run only once!
        //this.generateResTypeTable();
    }
    /**
     * 解析 xml 文档
     */
    public void readFile()
    {
        String str;
        int m=0;
        try {
            while(br.ready()){
                str=br.readLine();
                if(m>=20){System.out.println("m>=paraNum! paraNum is too small,
please set a larger one");break;}
                //识别匹配的资源名称
                if(str.indexOf( "METRIC NAME")>-1){
                    if(str.indexOf("cpu_nice")>-1){
                        dealStr(m, str);
                        m++;
                    }
                    else if (str.indexOf("cpu_idle") > -1) {
                        dealStr(m, str);
                        m++;
                    } else if (str.indexOf("cpu_num") > -1) {
                        dealStr(m, str);
                        m++;
                    } else if (str.indexOf("disk_free") > -1) {
                        dealStr(m, str);
                        m++;
                    }
    //提取服务器所需要的性能参数，篇幅所限这里仅以 cpu_nice、cpu_idle、cpu_num 和
    disk_free
    //为例，如需要提取其他参数，可以添加类似的代码
                }
    //处理一行字符串， 调用以下三个函数
    private void dealStr(int i,String str){
        int val1[]=readNAMEIndex( str);
```

```
        parameter[i].name=str.substring(val1[0], val1[1]);
        val1=readVALIndex(str);
        parameter[i].val=str.substring(val1[0], val1[1]);
        val1=readTYPEIndex(str);
        parameter[i].type=str.substring(val1[0], val1[1]);
        val1=readUNITSIndex(str);
        parameter[i].units=str.substring(val1[0], val1[1]);
        //System.out.println("name"+i+":"+parameter[i].name+"
val:"+parameter[i].val+"   type="+parameter[i].type+"   units:"+parameter[i].
units);
    }
    private int [] readNAMEIndex(String str1){
        int val1[]=new int [2];
        int i=str1.indexOf("NAME");
        val1[0]=i+6;
        val1[1]=str1.indexOf('"',val1[0]);
        return val1;
    }
    private int [] readVALIndex(String str1){
        int val1[]=new int [2];
        int i=str1.indexOf("VAL");
        val1[0]=i+5;//val1[0]指向数值的第一个值
        val1[1]=str1.indexOf('"', val1[0]);//val1[1]-1 指向数值的最后一个值
        return val1;
    }
    private int [] readTYPEIndex(String str1){
        int val1[]=new int [2];
        int i=str1.indexOf("TYPE");
        val1[0]=i+6;
        val1[1]=str1.indexOf('"',val1[0]);
        return val1;
    }
    private int [] readUNITSIndex(String str1){
        int val1[]=new int [2];
        int i=str1.indexOf("UNITS");
        val1[0]=i+7;
        val1[1]=str1.indexOf('"',val1[0]);
        return val1;
    }
```

📖**点评与拓展：** 解析 XML 文件涉及编写计算机程序方面的知识，本例以 Java 程序为例，实际上，不管利用何种语言，其步骤都与关键代码中展示的一样，读者只要稍有 Java 语言方面的知识辅以注释就不难读懂本例。

16.3.3　性能参数监控数据库

16.3.2 节的一段 Java 代码已经将性能参数的信息解析了出来，接下来就要将这些解析出来的信息存放到中心数据库中，可以通过两张表来存放所有服务器的性能参数。resourcetype 表，存放性能参数名称和单位，这张表静态存在，比如 disk_free 参数的单位是 GB；resources 表存放各服务器的所有性能参数及参数的数值。通过这两个表的连接操作，就能获得性能参数的名称是什么、单位是什么、具体的数值是多少等完整的信息。

建立并更新数据库通过 Java 程序执行 SQL 语句来实现，步骤如下。

在本地装载 MySQL 的 JDBC 驱动 mysql-connector-Java-3.1.14-bin.jar。

利用 JDBC 驱动通过数据库的 IP 地址、端口号、数据库所在服务器的地址以及用户名和密码与数据库建立连接。

建立会话，以便在本地向数据库传输 SQL 语句。

在本地创建合适的 SQL 语句，并发到数据库服务器，数据库服务器执行收到的 SQL 语句，结束后返回执行结果。

本书仍然用 Java 语言实现以上步骤所述的装载 JDBC 驱动、建立及更新数据库等操作，其完整代码参见本书附带光盘中的 ReadGmondXML.Java 文件，这里还是仅列出其中的关键代码加以解释。

```
    //exchange data with MySQL
    public void exchangeWithDB(){
        //db 是作为函数形参传入的数据库操作对象
        String sql="SELECT * FROM resources r WHERE r.IP='"+/*"172.18.14.34"*/
IP+"';";
        if(db.query(sql)){
           //Updata the resource infomation
             System.out.println("Update the resource infomation");
             for(int i=0;i<paraNum;i++){
                 if(parameter[i].name!=""){
    sql="UPDATE resources SET "+parameter[i].name+"='"+parameter[i].val+"'"+"
WHERE IP='"+IP+"';";
                     db.updata(sql);
                 }
             }
        }
        else {//Insert the resource infomation
```

```
            System.out.println("Insert the resource infomation");
            if(IP!=null && IP!="")
            {   sql="INSERT INTO resources (IP) VALUES ('"+IP+"');";
                db.updata(sql);
                for(int i=0;i<paraNum;i++){
                    if(parameter[i].name!=""){
                        sql="UPDATE  resources  SET  "+parameter[i].name+"='"+
parameter[i].val+"'"+"WHERE IP='"+IP+"';";
                        db.updata(sql);
                    }
                }
            }
        }
        Java.util.Date date=new Java.util.Date();
        sql="UPDATE  resources  SET  updatetime  ='"+date.toString()+"'  WHERE
IP='"+IP+"';";
        db.updata(sql);
    }
    public void generateResTypeTable(){
        for(int i=0;i<paraNum;i++){
            String sql="INSERT INTO resourcetype (name, unit) VALUES ('"
                +parameter[i].name+"', '"+parameter[i].units+"');";
            db.updata(sql);
        }
    }
    public String getIP() {
        return IP;
    }
    public void setIP(String ip) {
        IP = ip;
    }
    }
```

解释完程序的关键代码后，我们讲述 SQL 语句文件的构成和 MySQL 数据库的表字段和数据，以及如何利用 shell 脚本将 Java 程序封装，并通过 shell 脚本的层层调用实现对数据库的定时更新，即动态收集网络服务器的性能参数信息。

❶ 传送到数据库服务器的 SQL 语句一部分是由上述 Java 程序动态生成，另一部分有关建立 resources 和 resourcetype 表的 SQL 语句手工输入。图 16-17 显示了创建 resources 表的 SQL 语句，图中第一行判定是否已经存在 resources 表，如果存在则将 resources 表删除，重新创建；图 16-18 显示了创建 resourcetype 表的 SQL 语句，同样是首先判定是否已经存在

resourcettype 表，最后将静态的信息数据插入该表。

```
DROP TABLE IF EXISTS `resources`;
CREATE TABLE `resources` (
  `IP` varchar(20) NOT NULL default '',
  `cpu_nice` varchar(20) default NULL,
  `cpu_idle` varchar(20) default NULL,
  `cpu_speed` varchar(20) default NULL,
  `cpu_num` varchar(20) default NULL,
  `disk_free` varchar(20) default NULL,
  `disk_total` varchar(20) default NULL,
  `load_fifteen` varchar(20) default NULL,
  `load_five` varchar(20) default NULL,
  `load_one` varchar(20) default NULL,
  `machine_type` varchar(20) default NULL,
  `mem_buffers` varchar(20) default NULL,
  `mem_cached` varchar(20) default NULL,
  `mem_free` varchar(20) default NULL,
  `mem_shared` varchar(20) default NULL,
  `mem_total` varchar(20) default NULL,
  `mtu` varchar(20) default NULL,
  `os_name` varchar(20) default NULL,
  `os_release` varchar(20) default NULL,
  `swap_free` varchar(20) default NULL,
  `swap_total` varchar(20) default NULL,
  `updatetime` varchar(45) default NULL,
  `isCluster` tinyint(1) default '0'
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='All kinds of every node''s
```

图 16-17　创建 resources 表的 SQL 语句

```
DROP TABLE IF EXISTS `resourcetype`;
CREATE TABLE `resourcetype` (
  `name` varchar(20) character set latin1 NOT NULL default '',
  `unit` varchar(20) character set latin1 NOT NULL,
  PRIMARY KEY  USING BTREE (`name`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 ROW_FORMAT=DYNAMIC;

--
-- Dumping data for table `resourcetype`
--

/*!40000 ALTER TABLE `resourcetype` DISABLE KEYS */;
INSERT INTO `resourcetype` (`name`,`unit`) VALUES
 ('cpu_idle','%'),
 ('cpu_nice','%'),
 ('cpu_num','CPUs'),
 ('cpu_speed','MHz'),
 ('disk_free','GB'),
 ('disk_total','GB'),
 ('load_fifteen',''),
 ('load_five',''),
 ('load_one',''),
 ('machine_type',''),
 ('mem_buffers','KB'),
 ('mem_cached','KB'),
 ('mem_free','KB'),
 ('mem_shared','KB'),
 ('mem_total','KB'),
 ('os_name',''),
 ('os_release',''),
 ('swap_free','KB'),
 ('swap_total','KB');
/*!40000 ALTER TABLE `resourcetype` ENABLE KEYS */;
```

图 16-18　创建 resourcetype 并插入数据

❷ 编译执行上述的 Java 程序后，各服务器的性能参数就写入到 MySQL 服务器了，我们可以测试其效果：利用 shell 登录到 MySQL，输入下面 SQL 语句查询两台服务器的信息，如图 16-19 标注所示，显示出存放在数据库的两台服务器的 20 个性能参数的数值。

```
select * from resources where IP='172.18.12.178' or IP='172.18.12.181'
```

```
mysql> mysql> select * from resources where IP='172.18.12.178' or IP='172.18.12.181';
+---------------+----------+----------+-----------+---------+-----------+------------+--------------+-----------+----------+-
-------------+-------------+------------+----------+------------+-----------+------+---------+---------------------+--------
---+-------------+------------------------------+-----------+
| IP            | cpu_nice | cpu_idle | cpu_speed | cpu_num | disk_free | disk_total | load_fifteen | load_five | load_one |
machine_type | mem_buffers | mem_cached | mem_free | mem_shared | mem_total | mtu  | os_name | os_release          | swap_fr
ee | swap_total | updatetime                   | isCluster |
+---------------+----------+----------+-----------+---------+-----------+------------+--------------+-----------+----------+-
-------------+-------------+------------+----------+------------+-----------+------+---------+---------------------+--------
---+-------------+------------------------------+-----------+
| 172.18.12.178 | 0.0      | 60.6     | 2992      | 2       | 17.081    | 25.618     | NULL         | 0.01      | 0.70     |
x86          | 66456       | NULL       | 5532     | NULL       | 472516    | NULL | Linux   | 2.6.18-1.2798.fc6xen | 982640
   | 1048568     | Thu Jun 21 09:59:01 CST 2007 |         0 |
| 172.18.12.181 | 0.0      | 84.5     | 2793      | 2       | 17.087    | 76.389     | 0.26         | 0.05      | 0.03     |
x86          | 167380      | 341440     | 22964    | 0          | 513684    | NULL | Linux   | 2.6.18-1.2798.fc6   | 2031520
   | 1048568     | Thu Jun 14 22:03:53 CST 2007 |         0 |
+---------------+----------+----------+-----------+---------+-----------+------------+--------------+-----------+----------+-
-------------+-------------+------------+----------+------------+-----------+------+---------+---------------------+--------
---+-------------+------------------------------+-----------+
2 rows in set (0.00 sec)
```

图 16-19　resources 中的数据

使用下面命令可以查看 resourcetype 表中的所有数据，如图 16-20 所示。

```
select * from resourcetype
```

```
mysql> select * from resourcetype;
+--------------+------+
| name         | unit |
+--------------+------+
| cpu_idle     | %    |
| cpu_nice     | %    |
| cpu_num      | CPUs |
| cpu_speed    | MHz  |
| disk_free    | GB   |
| disk_total   | GB   |
| load_fifteen |      |
| load_five    |      |
| load_one     |      |
| machine_type |      |
| mem_buffers  | KB   |
| mem_cached   | KB   |
| mem_free     | KB   |
| mem_shared   | KB   |
| mem_total    | KB   |
| os_name      |      |
| os_release   |      |
| swap_free    | KB   |
| swap_total   | KB   |
+--------------+------+
19 rows in set (0.00 sec)
```

图 16-20　resourcetype 中的数据

❸ 为了方便上述的 Java 程序的部署，我们将所有 Java 类文件及 SQL 文件打包到一个文件夹，名字为 NodeExplorer。利用一个 shell 脚本 run.sh 来封装这些入口程序，也就是说只要执行 run.sh 这个脚本，就可以方便地生成 XML 文件、对其进行解析并将解析的结果存放到 MySQL 数据库，run.sh 脚本内容如图 16-21 所示。

```
[root@sec NodeExplorer]# vi run.sh
#!/bin/bash
cd /root/NodeExplorer
echo "run Gmond"
telnet localhost 8649 > /root/NodeExplorer/gmond_msg_1.txt

echo "run ReadGmondXML"
java -classpath /root/NodeExplorer:/root/NodeExplorer/mysql-connector-java-3.1.14-bin.jar
NodeExplorer /root/NodeExplorer/gmond_msg_1.txt
```

图 16-21　run.sh 脚本

第一处标注部分是生成服务器性能参数的 XML 文件，第二处标注部分是运行 Java 程序，导入连接数据库驱动，对 XML 文件解析，并将解析的结果存放到 MySQL 数据库。

❹ 为了实现服务器性能参数的动态更新，我们通过每个结点周期性执行 run.sh 脚本来实现，因为每执行一次 run.sh 脚本，gmond 就会重新读取一次服务器的性能参数，并重新存放到数据库中。

而定时执行某 shell 脚本可以使用 Linux 系统的调度进程 cron 来实现，系统命令 crontab 可以用来设定系统何时运行哪个程序。每一个用户都可以有一个 crontab 文件来保存调度信息，使用它运行任意一个 shell 脚本或某个命令，按每小时运行一次，或一周三次等。crontab 文件格式如下：分<>时<>日<>月<>星期<>要运行的命令，其中<>表示空格，写入 crontab 文件中就可实现程序定时周期执行。我们使用 “crontab -e” 命令设置 cron 调度表，写入如下两条语句：

```
* * * * * /root/NodeExplorer/run.sh
0 * * * * rm -rf /root/NodeExplorer/log.txt
```

表示每分钟执行一次 run.sh 脚本，并且每小时清空一次日志文件 log.txt，cron 设定如图 16-22 所示。

```
[root@sec NodeExplorer]# crontab -e
* * * * * /root/NodeExplorer/run.sh
0 * * * * rm -rf /root/NodeExplorer/log.txt
```

图 16-22　cron 设定

❺ 如图 16-23 所示，最后使用 install.sh 脚本对以上的所有脚本和程序作最后一道封装，执行 install.sh 脚本后，它首先启动 gmond 进程监控服务器的性能，然后启动 crontab 调度表，crontab 按照上一步的 cron 设定按时执行相关命令。

```
[root@sec NodeExplorer]# vi install.sh
#run Ganglia gmond
gmond


crontab
```

图 16-23　install.sh 脚本对所有脚本的封装

点评与拓展： 在此总结一下脚本、Java 程序的嵌套过程：install.sh 启动了 gmond 进程，并激活了 crontab 调度表；crontab 根据设定每分钟运行一次 run.sh 脚本；每运行一次 run.sh 脚本，run.sh 就从 gmond 处获得服务器的性能参数，利用 Java 程序对 XML 文件进行解析并将解析的结果写入 MySQL 数据库。

16.3.4　网络服务器性能参数的可视化

以上三小节中的所有工作归结为 install.sh 脚本，执行了 install.sh 脚本就可以成功地将网络服务器的性能参数写入 MySQL 数据库服务器的 resources 表中，并通过 Linux 系统提供的 cron 功能每分钟更新一次。本节的工作就是要将所收集到 resources 表中的信息以 Web 页面的形式展示给用户，利用 Eclipse 开发工具生成.war 文件，来读取数据库信息，再以 JSP 网页图形化显示各服务器的性能参数。利用 Eclipse 开发 Web 工程的步骤在第 8 章“Web 服务器的配置与架设”中详细讲过，本节不再详述，而只是讲述其中的 JSP 关键代码，以及图形化显示数据的相关知识，最后演示所开发的网络服务器监控系统。

1. 在 JSP 页面上显示数据库表信息可以分为两步，第一步是连接 MySQL 数据库服务器，提交查询 resources 表的 SQL 语句以读取 resources 表中所有数据记录；第二步是将上一步中读到的数据记录在 JSP 网页上显示出来。

   第一步的所有操作我们用一个 JavaBean 进行封装。JavaBean 是用 Java 语言写成的可重用组件，可以一次性编写，供 JSP 调用。第一次接触 JavaBean 的读者可以将它看作是一个封装好的 Java 类，JSP 调用这个 JavaBean 时，传入所要执行的 SQL 语句，就可以自动实现连接数据库、提交该 SQL 语句到数据库服务器，并将查询结果返回 JSP。JavaBean 又利用 DBManager 类来实现连接数据库、执行 SQL 语句的功能：DBManager 类的构造函数用于连接 MySQL 数据库，图 16-24 显示了它的关键代码，图中第一处标注处是下载连接数据库的驱动程序，在有了驱动程序的基础上，就可以使用 getConnection 函数连接指定的 MySQL 数据库，如图 16-24 第二处标注部分所示。

   DBManager 类中另一个关键函数名为 queryResourceInfobyIP。我们知道 resources 表以 IP 地址来标识每个数据记录，JSP 显示也是逐台主机显示其性能参数信息的，因此，显示服务器的性能参数就是以 IP 为查询关键字将其他字段的数据取出。queryResourceInfobyIP 类就实现了由 IP 为关键字获得所有性能参数的功能，它的入口参数为 IP 地址。图 16-25 显示了 queryResourceInfobyIP 的代码，图中第一处标注处是根据用户提交的 IP 地址生成 SQL 语句；

第二处标注部分为所生成的 SQL 语句提交到 MySQL 数据库服务器执行。DBManager 类的完整代码参见本书附带光盘的 DBManager.Java 文件，我们对该类的其他函数也提供了一定的注释以帮助读者理解。

```
/**
 * @author HuangHe
 *
 */
public class DBmanager {

    private Connection con;

    private Statement st;

    private ResultSet rs=null;

    private String[] columnNames=null;
    private String[] columnUnits=null;

    public DBmanager() {
        super();
        try {
            Class.forName("com.mysql.jdbc.Driver");
            System.out.println("\n\nSuccess loading mysql Driver....");
        } catch (ClassNotFoundException e) {
            System.err.println("\n\nError loading mysql Driver.....");
            e.printStackTrace();
            System.exit(1);
        }

        try {//连接数据库
            con=DriverManager.getConnection("jdbc:mysql://172.18.12.179:3306/seugrid","seugrid","seugrid");
//          con=DriverManager.getConnection("jdbc:mysql://localhost:3306/seugrid","root","seugrid");
            st=con.createStatement();
            System.out.println("Success loading mysql Database...");
        } catch (SQLException e) {
            System.err.println("\n\n\nConnecting mysql database fail!\n" +
                    "Please check up whether mysql database is startup " +
                    "and whether the  Database's address and account is correct." +
                    " \n\n\n");
            e.printStackTrace();
        }
    }
```

图 16-24　DBManager 类构造函数的代码

```
/**
 * 通过IP地址查询局域网中某服务器的资源信息,
 *入口参数为要查节点的IP地址.
 *返回包含所有资源值的字符串.
 *
 */
    public String [] queryResourceInfoByIP(String IP){
        String []queryStr=null;
        String sql="SELECT * FROM resources r WHERE IP='"+IP+"';";
        System.out.println("sql statement: "+sql);
        try {
            rs=st.executeQuery(sql);
            ResultSetMetaData rsmd=rs.getMetaData();
            int n=rsmd.getColumnCount();
            if(rs.next()){//if result set is not empty
            //fill column name and column unit if they are null
                if(columnNames==null){
                    columnNames=new String[n];
                    columnUnits=new String[n];
                    for(int i=0;i<n;i++){
                        columnNames[i]=rsmd.getColumnName(i+1);
                        String sql1="SELECT r.`unit` FROM seugrid.resourcetype r  WHERE name='"
                            +columnNames[i]+"';";
                        rs=st.executeQuery(sql1);
                        if(rs.next())
                            columnUnits[i]=rs.getString(1);
                        else columnUnits[i]=new String("");
                    }
                }
                rs=st.executeQuery(sql);
                queryStr=new String[n];
                while(rs.next()){
                    for(int i=0;i<n;i++){
                        queryStr[i]=rs.getString(i+1);
                    }
                }

            }
            //else return null;
        } catch (SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        return queryStr;
    }
```

图 16-25　queryResourceInfobyIP 函数的代码

❷ resourceInfo.jsp 作为 Web 工程下面的子文件，负责调用 JavaBean 获得指定 IP 服务器的性能参数，并分别以文字的形式和图形的形式显示。图 16-26 显示了 resourceInfo.jsp 用于查询数据库并以文字形式显示性能参数的代码，第一处标注的代码为向 DBManager 提交查询 SQL 语句，返回的结果保存在 resourceInfo 数组中。大括号标注部分表示将 resourceInfo 数组显示在 Web 页面上。

以图形化显示性能参数信息，我们使用 JFreeChart 来实现。JFreeChart 是 Java 平台上的一个开放的图表绘制类库，它完全使用 Java 语言编写，是为 Java applications、applets、servlets 以及 JSP 等使用所设计的。JFreeChart 可生成饼图(pie charts)、柱状图(bar charts)、散点图(scatter plots)、时序图(time series)、甘特图(Gantt charts)等多种图表，并且可以产生 PNG 和 JPEG 格式的输出。图 16-27 显示了利用 JFreeChart 绘制磁盘信息饼图的代码，其中的标注部分是 JFreeChart 获取磁盘数据的函数。

```
<%
    IP = request.getParameter("IP");
    if (IP != null) {
        resourceInfo = database.queryResourceInfoByIP(IP);
        if (resourceInfo != null) {
            columnNames = database.getColumnNames();
            columnUnits = database.getColumnUnits();
            out.println("<p>The <font color=\"red\">red font</font> information is dynamic
            resource.</p><p> </p>");
            for (int i = 0; i < resourceInfo.length; i++) {
            out.println("<p>" );
            如果是动态资源信息,将字体标为红色
            if(i==2||i==5||i==7||i==8||i==9||i==13||i==19)
                out.println("<font color=\"red\">");
            out.println(columnNames[i] + ":  "
                + resourceInfo[i] + "  "
                + columnUnits[i] );
            if(i==2||i==5||i==7||i==8||i==9||i==13||i==19)
                out.println("</font>");
            out.println("</p>");
            }
        }
        else out.println("<p>IP address: "+IP+" does not exist, please check your input!"+"</p>");

    }
    else out.println("<p>You do not input any IP address, please input an IP address!</p>");
%>
```

图 16-26　resourceInfo.jsp 显示性能参数部分代码

```
<p>Visual Display</p>
<!--数据可视化代码, Disk 饼图-->
<%
    String titlel="Disk Volumn";
    DefaultPieDataset piedatal=new DefaultPieDataset();
    piedatal.setValue("disk_used",Double.parseDouble(resourceInfo[6])-Double.parseDouble(resourceInfo[5]));
    piedatal.setValue("disk_free",Double.parseDouble(resourceInfo[5]));

    JFreeChart chartl =ChartFactory.createPieChart(titlel, piedatal, true, true, true);
    chartl.setTitle(new TextTitle(titlel,new Font("New Times Roman", Font.ITALIC, 15)));

    PiePlot plotl=(PiePlot) chartl.getPlot();

    //ChartRenderingInfo info=new ChartRenderingInfo(new StandardEntityCollection());
    String fileNamel=ServletUtilities.saveChartAsJPEG(chartl, 400,300, null,session);
    String urll=request.getContextPath()+"/servlet/DisplayChart?filename="+fileNamel;
    System.out.println("url:"+urll);
%>
```

图 16-27　利用 JFreeChart 图形化显示部分代码

❸ 将完整的.war 文件部署到 Tomcat 服务器后，首页提供各网络服务器的查询功能，如图 16-28 所示。输入 172.18.12.178，单击 Submit Query 按钮，弹出如图 16-29 所示的界面，窗口左侧以文字形式显示了 172.18.12.178 这台服务器的性能参数，其中红字显示的参数为动态参数；窗口右侧绘制了磁盘、内存和交换区的可用空间与剩余空间之间关系的饼图。

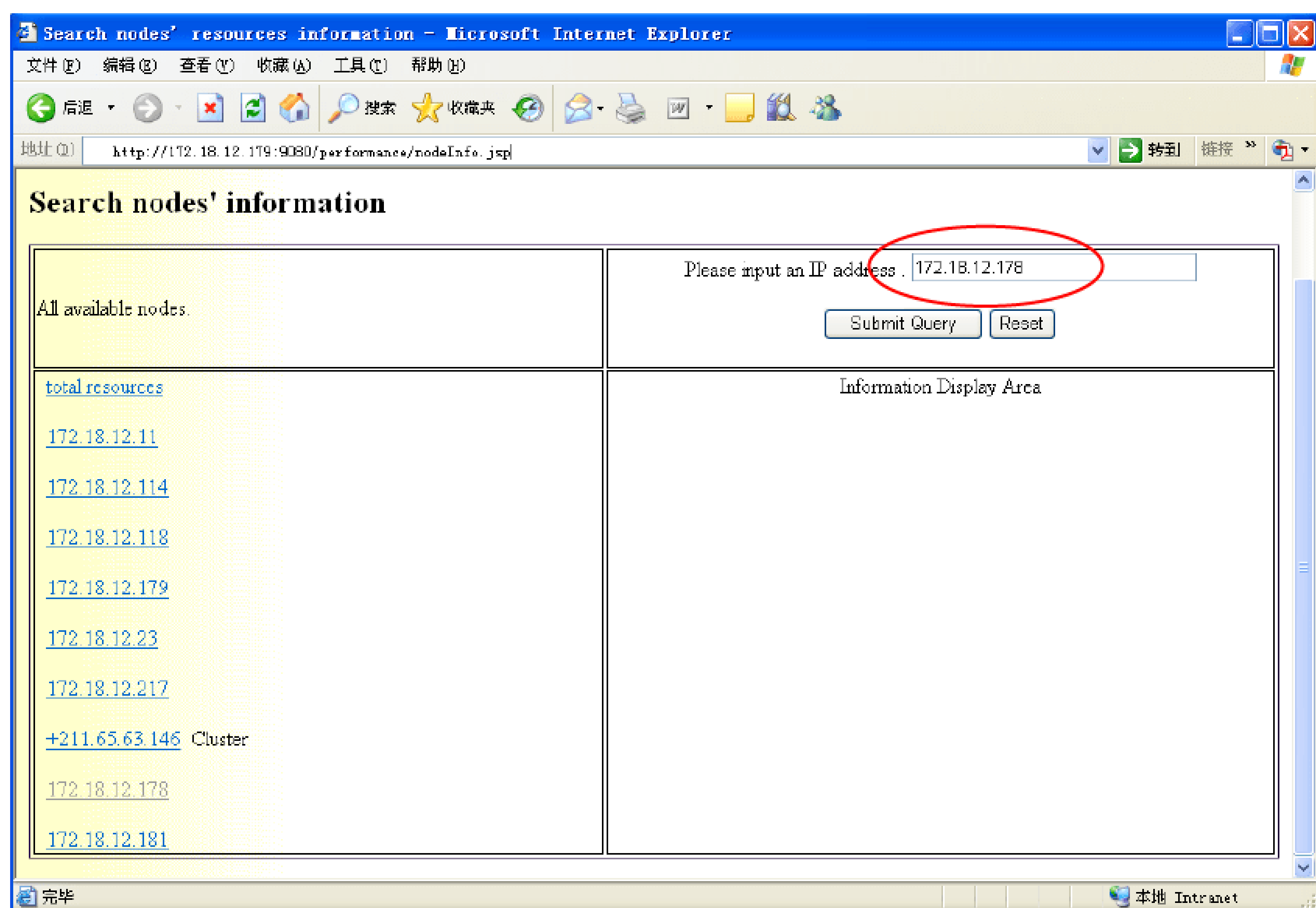


图 16-28　服务器查询

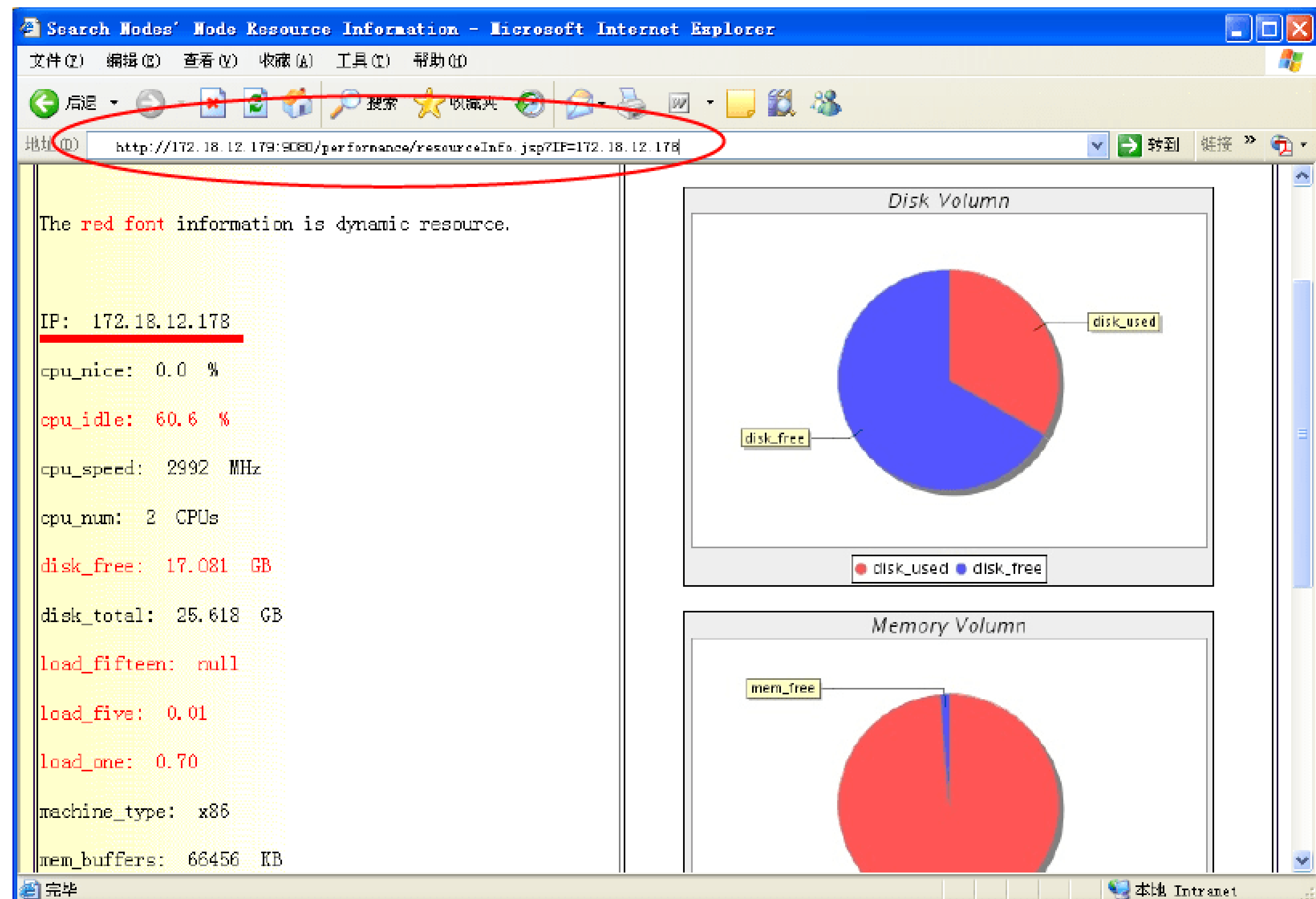


图 16-29　显示网络服务器的性能参数

📖**点评与拓展：** 本节讲述了利用 JavaBean 和 JSP 技术实现网页与数据库的交互，着重讲述了 JavaBean、JSP 代码内模块的意义，开发 JSP 和部署 Web 工程到 Tomcat 的步骤与第 8 章中相同。

16.4 本 章 小 结

本章首先介绍了网络服务器监控的意义、特点和难点，以及国际上的研究与监控系统实现的现状，指出网络服务器监控已经成为架设 Linux 服务器必须要面对的重要问题。接着，本章以当今分布式计算中的服务器监控案例为背景，结合 Ganglia、MySQL、shell 脚本、Java、Web 服务器及 JSP 开发等前面章节的知识和实用技术，实现了一个较为复杂的网络服务器性能监控系统，重点讲述了整个系统中功能模块的划分，以及每个模块利用何种技术来实现，并对各种技术中的关键部分作了解释。实际上，实现某些功能模块所用到的技术可能不是惟一的，但是设计和开发大型系统的思路却是一致的。

参 考 文 献

1. 谢希仁. 计算机网络(第 4 版). 北京：电子工业出版社，2004

2. Richard Stevens W 著. 范建华，胥光辉，张涛等译. TCP/IP 详解 卷 Ⅰ：协议. 北京：机械工业出版社，2005

3. 吴学毅主编. Linux 基础教程. 北京：清华大学出版社，2005

4. 赵敬明编译. shell 命令语言及程序设计. 北京：电子工业出版社，2000

5. 林慧琛，刘殊，尤国君等编著. Red Hat Linux 服务器配置与应用. 北京：人民邮电出版社，2006

6. 鸟哥的 Linux 私房菜. http://linux.vbird.org

7. Grace M. Buechlein. Linux Complete Command Reference .U.S.: Redhat Press,1997

8. HalStern, Mike Eisler,Ricardo Labiaga. Managing NFS and NIS 2nd edition. U.S.: O'Reilly，

9. Gilfillan I 著. 王军等译. Mastering MySQL4. 北京：电子工业出版社，2003

10. Ganglia——distributed monitoring system. http://ganglia.sourceforge.net

11. JFreeChart. http://www.jfree.org/jreechart